
HTTP/2 & QUIC
TEACHING GOOD PROTOCOLS TO DO BAD THINGS

PEOPLE - KATE

- Catherine (Kate) Pearce
  - @secvalve
- Sr. Security Consultant (Customer Focused) at Cisco
  - Break & report
  - Coach the builders
  - Research what's ahead
- Distinguishing Features:
  - Loud, Yellow
  - Or is that "Loud Yellow"?

PEOPLE - KATE

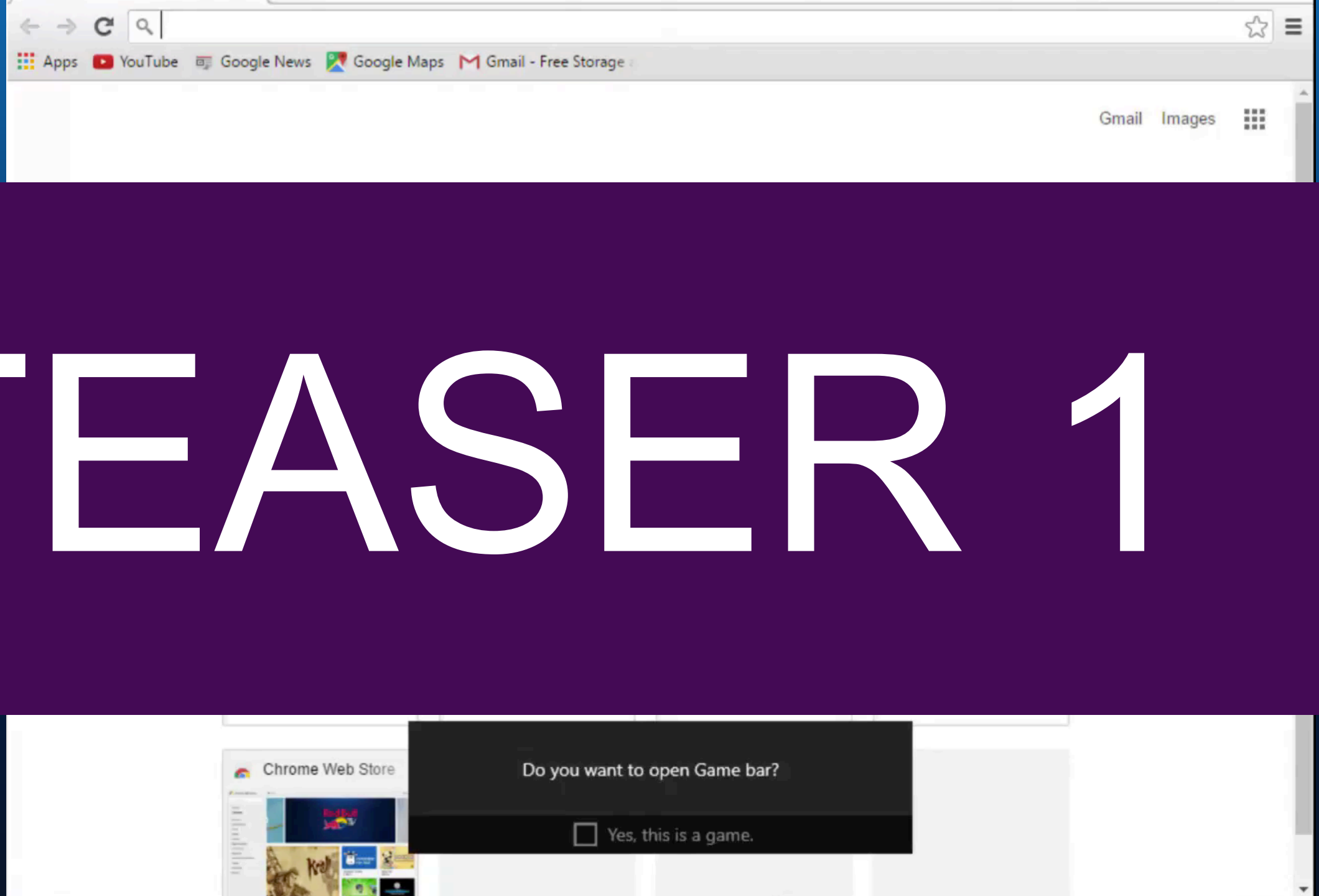
- Plays with fire, will never have a better photo taken in her life:

CCSA, https://m.flikr.com/#/photos/4nitsirk

PEOPLE - VYRUS

- Carl Vincent
  - Security Consultant

- Distinguishing Features:
  - Hates photos
  - Red team guy
  - Jack of many trades, in search of more!
  - Suffers from a severe compulsion to continually contemplate the best way to control, and/or destroy, absolutely everything and everyone in the room – including the room itself.


Apps
YouTube
Google News
Google Maps
Gmail - Free Storage
Gmail
Images
Chrome Web Store
Do you want to open Game bar?
Yes, this is a game.

TEASER 1

Teaser 1

Wait… firewall was blocking ALL TCP?

Teaser 1

Wireshark · Protocol Hierarchy Statistics · demo_video_win_10_quic

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes
Frame	100.0	9511	100.0	8361781	540 k	0	0
Ethernet	100.0	9511	100.0	8361781	540 k	0	0
Internet Protocol Version 6	0.1	7	0.0	915	59	0	0
User Datagram Protocol	0.0	1	0.0	153	9	0	0
DHCPv6	0.0	1	0.0	153	9	1	153
Internet Control Message Protocol v6	0.1	6	0.0	762	49	6	762
Internet Protocol Version 4	99.7	9478	100.0	8359708	540 k	0	0
User Datagram Protocol	99.7	9478	100.0	8359708	540 k	0	0
Teredo IPv6 over UDP tunneling	0.1	6	0.0	762	49	0	0
QUIC (Quick UDP Internet Connections)	98.5	9365	99.7	8337858	538 k	9365	8337858
Domain Name System	1.1	107	0.3	21088	1362	107	21088
Address Resolution Protocol	0.3	32	0.0	1920	124	32	1920

Teaser 1

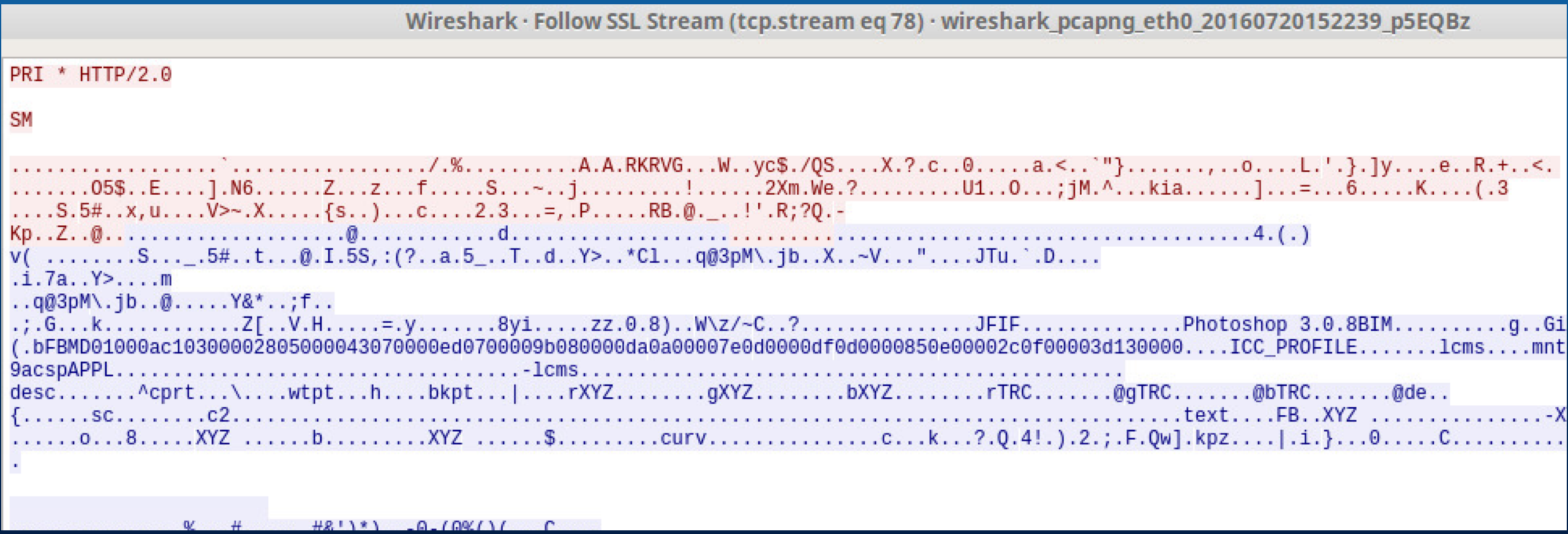
```
User Datagram Protocol, Src Port: 63786 (63786), Dst Port: 443 (443)
QUIC (Quick UDP Internet Connections)
 Public Flags: 0x0d
 CID: 14646921831679203 67
 Version: Q030
 Sequence: 1
 Message Authentication Hash: 870608f6bf34b710f976e324
 Private Flags: 0x00
 STREAM (Special Frame Type) Stream ID:1, Type: CHLO (Client Hello)
  Frame Type: STREAM (Special Frame Type) (0xa0)
  Stream ID: 1
  Data Length: 1024
  Tag: CHLO (Client Hello)
  Tag Number: 27
  Padding: 0000
  Tag/value: PAD (Padding) (l=292)
  Tag/value: SNI (Server Name Indication) (l=16): www.google.co.nz
  Tag/value: STK (Source Address Token) (l=58)
  Tag/value: VER (Version) (l=4) Q030
  Tag/value: CCS (Common Certificate Sets) (l=16)
  Tag/value: NONC (Client Nonce) (l=32)
  Tag/value: MSPC (Max streams per connection) (l=4): 100
  Tag/value: AEAD (Authenticated encryption algorithms) (l=4), AES-GCM with a 12-byte tag and IV
```

Teaser 1

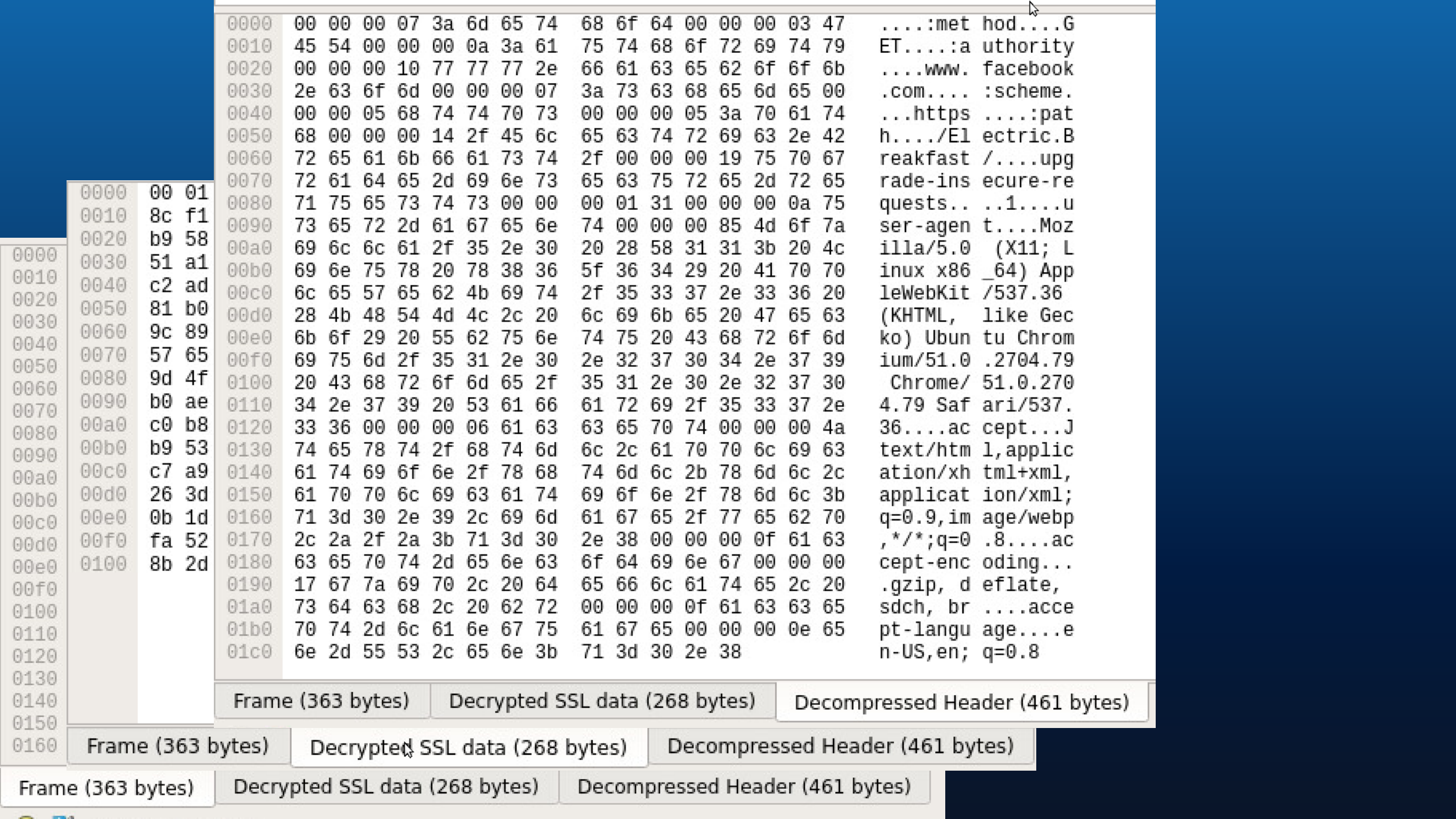
```
Tag/value: AEAD (Authenticated encryption algorithms) (l=4), AES-GCM with a 1
Tag/value: UAID (Client's User Agent ID) (l=50): m Chrome/51.0.2704.106 Windo
Tag/value: SCID (Server config ID) (l=16)
Tag/value: TCID (Connection ID truncation) (l=4)
Tag/value: PDMD (Proof Demand) (l=4): X509
Tag/value: SRBF (Socket receive buffer) (l=4)
Tag/value: ICSL (Idle connection state) (l=4)
Tag/value: CTIM (Unknown) (l=8)
Tag/value: NONP (Unknown) (l=32)
Tag/value: PUBS (Public value) (l=32)
Tag/value: SCLS (Silently close on timeout) (l=4)
Tag/value: KEXS (Key exchange algorithms) (l=4), Curve25519
Tag/value: XLCT (Unknown) (l=8)
Tag/value: CSCT (Unknown) (l=0)
Tag/value: COPT (Connection options) (l=4)
Tag/value: CCRT (Cached certificates) (l=24)
Tag/value: IRTT (Estimated initial RTT) (l=4): 240909
Tag/value: CETV (Client encrypted tag-value) (l=164)
Tag/value: CFCW (Initial session/connection) (l=4): 15728640
Tag/value: SFCW (Initial stream flow control) (l=4): 6291456
```

Teaser 2

What type of traffic is this?



What type of traffic is this?


0000 00 00 00 07 3a 6d 65 74 68 6f 64 00 00 00 03 47 ....:met hod....G
0010 45 54 00 00 00 0a 3a 61 75 74 68 6f 72 69 74 79 ET....:a uthority
0020 00 00 00 10 77 77 77 2e 66 61 63 65 62 6f 6f 6b ....www. facebook
0030 2e 63 6f 6d 00 00 00 07 3a 73 63 68 65 6d 65 00 .com.... :scheme.
0040 00 00 05 68 74 74 70 73 00 00 00 05 3a 70 61 74 ...https ....:pat
0050 68 00 00 00 14 2f 45 6c 65 63 74 72 69 63 2e 42 h..../El ectric.B
0060 72 65 61 6b 66 61 73 74 2f 00 00 00 19 75 70 67 reakfast /....upg
0070 72 61 64 65 2d 69 6e 73 65 63 75 72 65 2d 72 65 rade-ins ecure-re
0080 71 75 65 73 74 73 00 00 00 01 31 00 00 00 0a 75 quests.. ..1....u
0090 73 65 72 2d 61 67 65 6e 74 00 00 00 85 4d 6f 7a ser-agen t....Moz
00a0 69 6c 6c 61 2f 35 2e 30 20 28 58 31 31 3b 20 4c illa/5.0 (X11; L
00b0 69 6e 75 78 20 78 38 36 5f 36 34 29 20 41 70 70 inux x86 _64) App
00c0 6c 65 57 65 62 4b 69 74 2f 35 33 37 2e 33 36 20 leWebKit /537.36
00d0 28 4b 48 54 4d 4c 2c 20 6c 69 6b 65 20 47 65 63 (KHTML, like Gec
00e0 6b 6f 29 20 55 62 75 6e 74 75 20 43 68 72 6f 6d ko) Ubun tu Chrom
00f0 69 75 6d 2f 35 31 2e 30 2e 32 37 30 34 2e 37 39 ium/51.0 .2704.79
0100 20 43 68 72 6f 6d 65 2f 35 31 2e 30 2e 32 37 30 Chrome/ 51.0.270
0110 34 2e 37 39 20 53 61 66 61 72 69 2f 35 33 37 2e 4.79 Saf ari/537.
0120 33 36 00 00 00 06 61 63 63 65 70 74 00 00 00 4a 36....ac cept...J
0130 74 65 78 74 2f 68 74 6d 6c 2c 61 70 70 6c 69 63 text/htm l,applic
0140 61 74 69 6f 6e 2f 78 68 74 6d 6c 2b 78 6d 6c 2c ation/xh tml+xml,
0150 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78 6d 6c 3b applicat ion/xml;
0160 71 3d 30 2e 39 2c 69 6d 61 67 65 2f 77 65 62 70 q=0.9,im age/webp
0170 2c 2a 2f 2a 3b 71 3d 30 2e 38 00 00 00 0f 61 63 ,*/*;q=0 .8....ac
0180 63 65 70 74 2d 65 6e 63 6f 64 69 6e 67 00 00 00 cept-enc oding...
0190 17 67 7a 69 70 2c 20 64 65 66 6c 61 74 65 2c 20 .gzip, d eflate,
01a0 73 64 63 68 2c 20 62 72 00 00 00 0f 61 63 63 65 sdch, br ....acce
01b0 70 74 2d 6c 61 6e 67 75 61 67 65 00 00 00 0e 65 pt-langu age....e
01c0 6e 2d 55 53 2c 65 6e 3b 71 3d 30 2e 38 n-US,en; q=0.8
Frame (363 bytes)
Decrypted SSL data (268 bytes)
Decompressed Header (461 bytes)
0000 00 01
0010 8c f1
0020 b9 58
0030 51 a1
0040 c2 ad
0050 81 b0
0060 9c 89
0070 57 65
0080 9d 4f
0090 b0 ae
00a0 c0 b8
00b0 b9 53
00c0 c7 a9
00d0 26 3d
00e0 0b 1d
00f0 fa 52
0100 8b 2d
Frame (363 bytes)
Decrypted SSL data (268 bytes)
Decompressed Header (461 bytes)
Frame (363 bytes)
Decrypted SSL data (268 bytes)
Decompressed Header (461 bytes)

What's going on here?

→*Let's talk about ~~upcoming~~ web transport protocols*

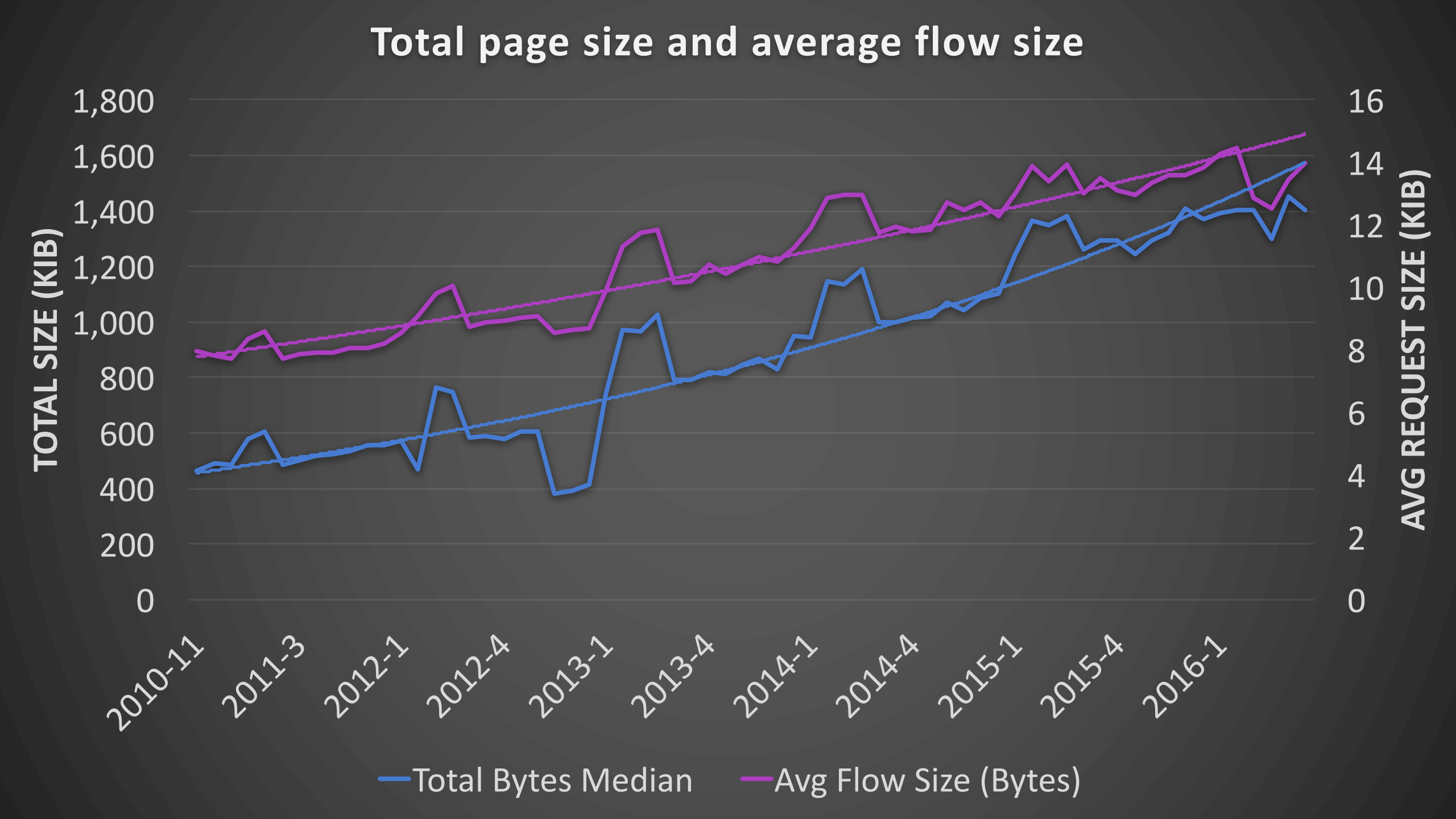
What's going on here?

→*Let's talk about recent web transport protocols*

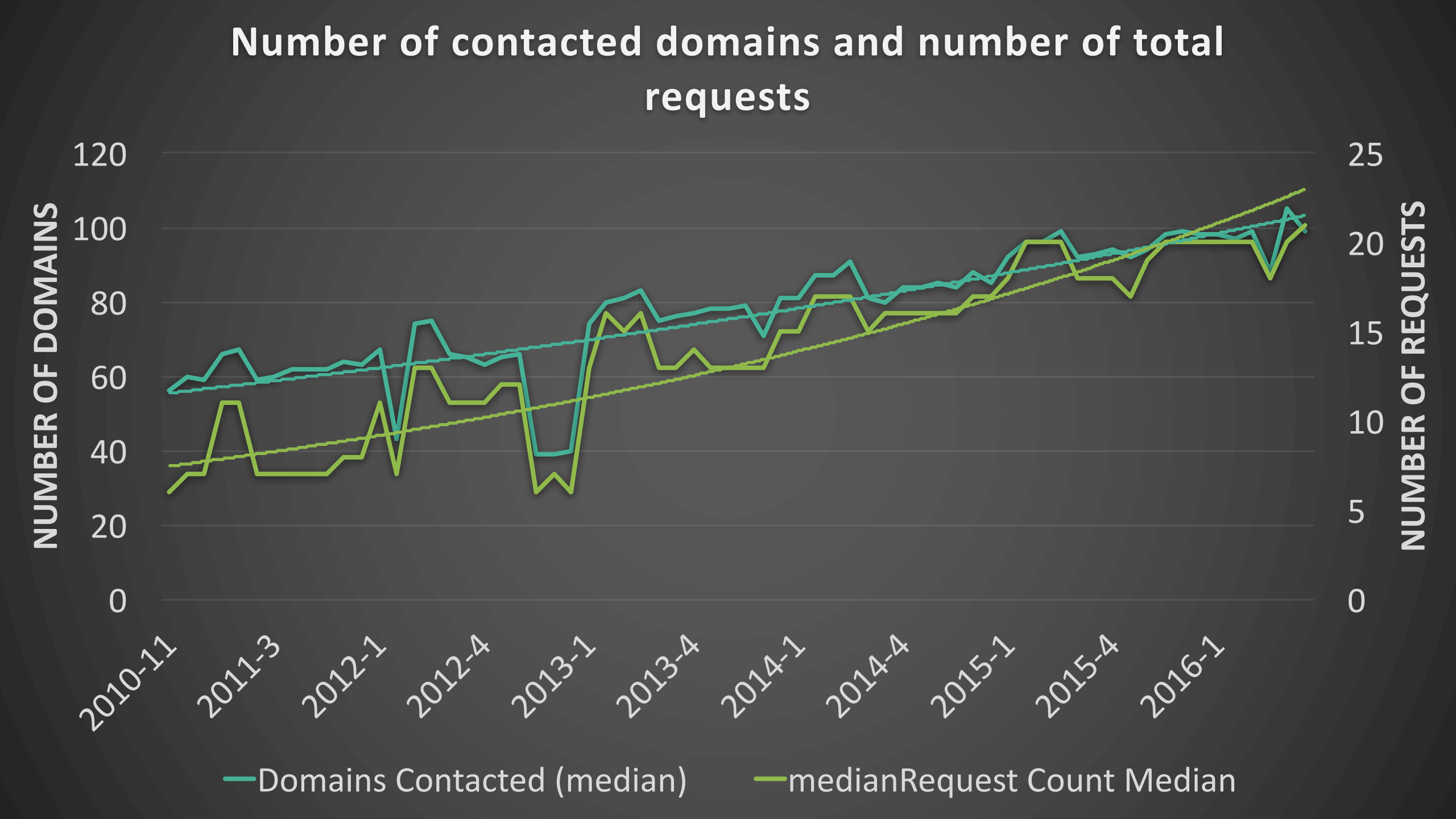

INTRO
(WHY IS THE WORLD EXPLODING?)

DRIVERS FOR CHANGE

- Increasing scale of…everything
  - Flow size increases
  - Flow count increases (e.g. web pages)
  - Flow diversity increases (e.g. web pages)
    - Mobility
    - Multiple connections


Total page size and average flow size
TOTAL SIZE (KIB)
AVG REQUEST SIZE (KIB)
1,800
1,600
1,400
1,200
1,000
800
600
400
200
0
16
14
12
10
8
6
4
2
0
2010-11
2011-3
2012-1
2012-4
2013-1
2013-4
2014-1
2014-4
2015-1
2015-4
2016-1
Total Bytes Median
Avg Flow Size (Bytes)

Number of contacted domains and number of total requests



WHY IS THIS HAPPENING?

Network communication needs better capabilities, but there's more than one way to do it

1. HTTP/2 - Multiplexes within TCP
2. QUIC - Ignores TCP to handle it itself

These technologies change the way the internet behaves

WHY DO YOU CARE?

Familiar Problems

- Opaque Technology Shifts

“New” Problems

- New Fragmentation Attacks
- Blind Network Security

TO BE CLEAR:

These technologies are more culture shock than direct vulnerabilities / concerns

Personally, we like them, and want them to succeed

Network tools and operators need to be ready

I’m skipping ENORMOUS amounts of detail.

BACKGROUND

(HOW DID WE GET HERE)

PREVIOUS WORK

- MPTCP
- MPTCP Implications
- Multipath Implications
- Multipath “defences”

WHY NOT CHANGE TCP?

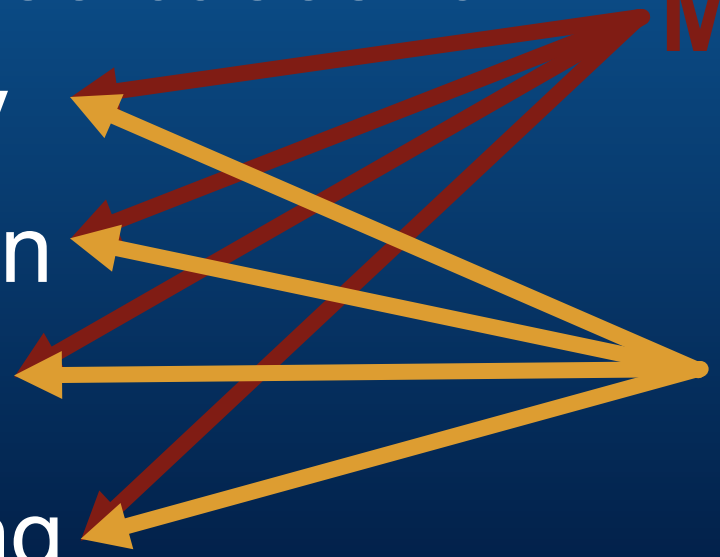
Lessons from MPTCP:

- Slow moving, OS- and hardware-dependent
- Middleboxes limit protocol deployability
- Chicken and egg deployment

CURRENT TCP IS RATHER LIMITED

Doesn’t support use cases for:

- High Availability
- Link Aggregation
- Multihoming
- Mesh networking



Makes a lot of round trips

Blocks stream on retransmits

WHY NOT CHANGE TCP?

WHY NOT CHANGE TCP?

TCP Characteristics:

- Handshake design
- Outside user-space
- End-of-line blocking

WHY NOT CHANGE TCP?

If you can't change TCP, what's left?

- SCTP?
  - Same problems, but amplified
- Application Layer?
  - Http/2 & SPDY
- UDP?
  - But it doesn't do ANYTHING fancy?
  - Exactly – QUIC

BACKGROUND – THE JOURNEY TO HERE

TCP -> MPTCP -> QUIC

BACKGROUND – THE JOURNEY TO HERE

HTTP -> SPDY -> HTTP/2

SO WHAT?

- Have you realized how many security tools support these?
  - It’s… unfortunate

REAL-WORLD PREVALENCE

- MPTCP developed surprisingly fast, then faltered
- QUIC was even QUIC-ker
  - Already in use on many Google properties
    - Youtube, Google search, and more
  - Likely several percent of your traffic
- Http/2 has become real-world even faster

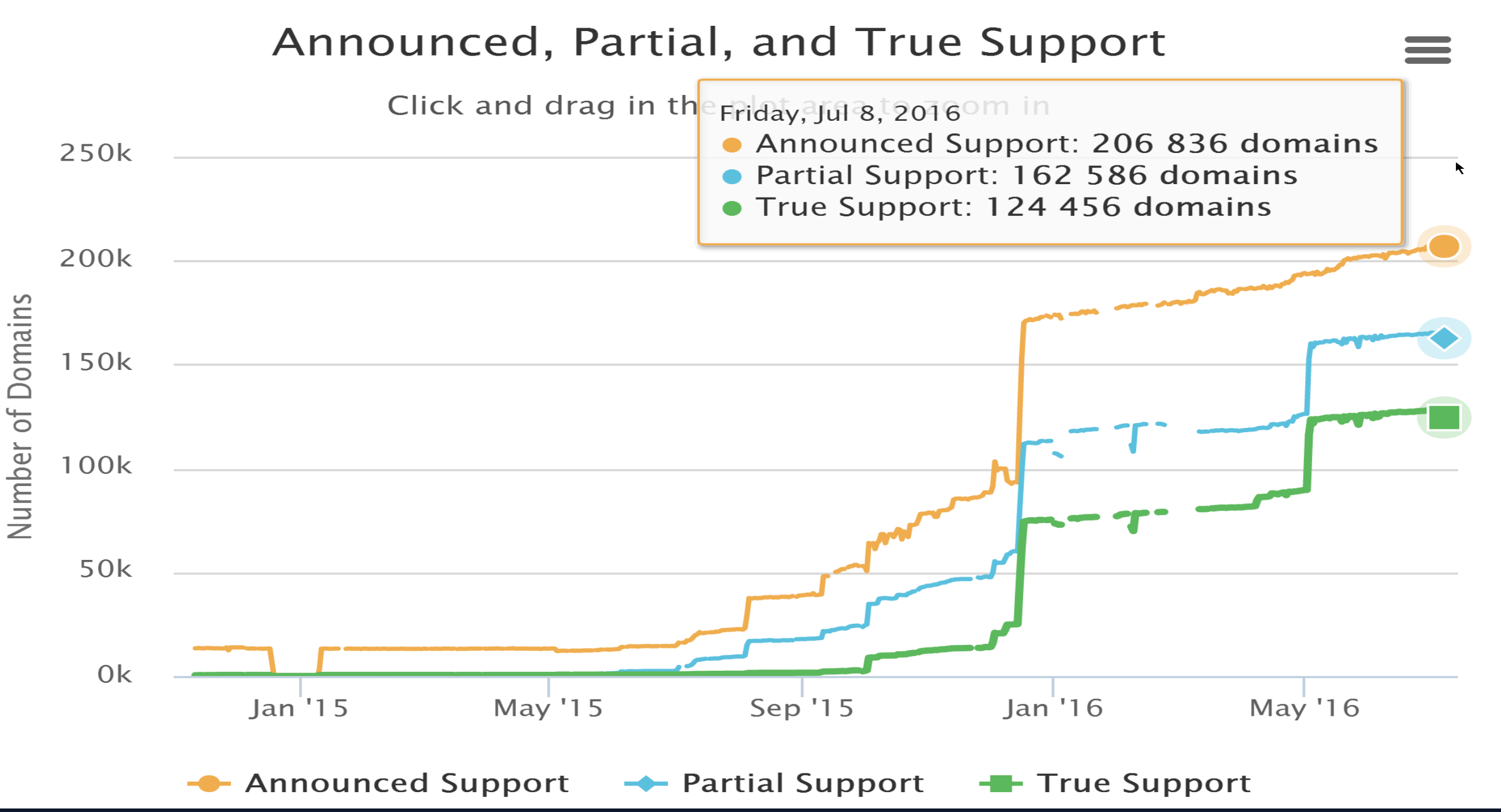
PROTOCOL PREVALENCE

	Servers	Clients	Key usages
MPTCP	~5000?	50 000 000	Apple iOS (Siri), OVF OverTheBox
QUIC			
HTTP2			

PROTOCOL PREVALENCE

	Servers	Clients	Key usages
MPTCP	~5000?	50 000 000	Apple iOS (Siri), OVF OverTheBox
QUIC	~25000 [2]	1 000 000 000+[1]	Google Chrome, Google Duo, Google Websites
HTTP2			

1 - https://chrome.googleblog.com/2016/04/chrome-50-releases-and-counting.html
2 - https://www.shodan.io/



1 - http://isthewebhttp2yet.com/measurements/adoption.html#time

PROTOCOL PREVALENCE

	Servers	Clients	Key usages
MPTCP	~5000?	50 000 000	Apple iOS (Siri), OVF OverTheBox
QUIC	~25000 [2]	1 000 000 000+[1]	Google Chrome, Google Duo, Google Websites
HTTP2	200 000+ [3]	~2 000 000 000 [4]	Chrome, Edge, Firefox Twitter, Facebook, Yahoo, Google

1 - https://chrome.googleblog.com/2016/04/chrome-50-releases-and-counting.html
2 - Shodan
3 - http://isthewebhttp2yet.com/measurements/adoption.html#time
4 - Uncertain, every up-to-date popular browser supports it

REAL-WORLD PREVALENCE

```
username@bhubu ~/scanning $ head -n 20 hosts.txt | xargs -P 40 -I {} -i bash -c 'echo -e $(is-http2 www.{} | t
✓ HTTP/2 supported by www.facebook.com Supported protocols: h2 h2-fb spdy/3.1-fb-0.5 spdy/3.1 spdy/3 http/1.1
× HTTP/2 not supported by www.baidu.com Supported protocols: http/1.1
× HTTP/2 not supported by www.bing.com
✓ HTTP/2 supported by www.google.co.in Supported protocols: h2 spdy/3.1 http/1.1
× HTTP/2 not supported by www.msn.com
✓ HTTP/2 supported by www.twitter.com Supported protocols: h2 spdy/3.1 http/1.1
✓ HTTP/2 supported by www.google.co.jp Supported protocols: h2 spdy/3.1 http/1.1
× HTTP/2 not supported by www.qq.com Supported protocols: http/1.1 http/1.0
✓ HTTP/2 supported by www.wikipedia.org Supported protocols: h2 http/1.1
✓ HTTP/2 supported by www.google.com Supported protocols: h2 spdy/3.1 http/1.1
× HTTP/2 not supported by www.amazon.com Supported protocols: http/1.1
× HTTP/2 not supported by www.linkedin.com Supported protocols: spdy/3.1 spdy/3 http/1.1 http/1.0
× HTTP/2 not supported by www.vk.com Supported protocols: spdy/3.1 http/1.1
✓ HTTP/2 supported by www.yahoo.com Supported protocols: h2 h2-14 spdy/3.1 spdy/3 http/1.1 http/1.0
✓ HTTP/2 supported by www.youtube.com Supported protocols: h2 spdy/3.1 http/1.1
× HTTP/2 not supported by www.live.com
× HTTP/2 not supported by www.sina.com.cn
× HTTP/2 not supported by www.taobao.com Supported protocols: spdy/3.1 http/1.1
✓ HTTP/2 supported by www.instagram.com Supported protocols: h2 h2-fb http/1.1
```

9 of 19 Alexa Top Sites support H2 or SPDY

ABOUT

(WHAT'S IN FRONT OF US, AND HOW DO THESE WORK?)

COMMON GOALS

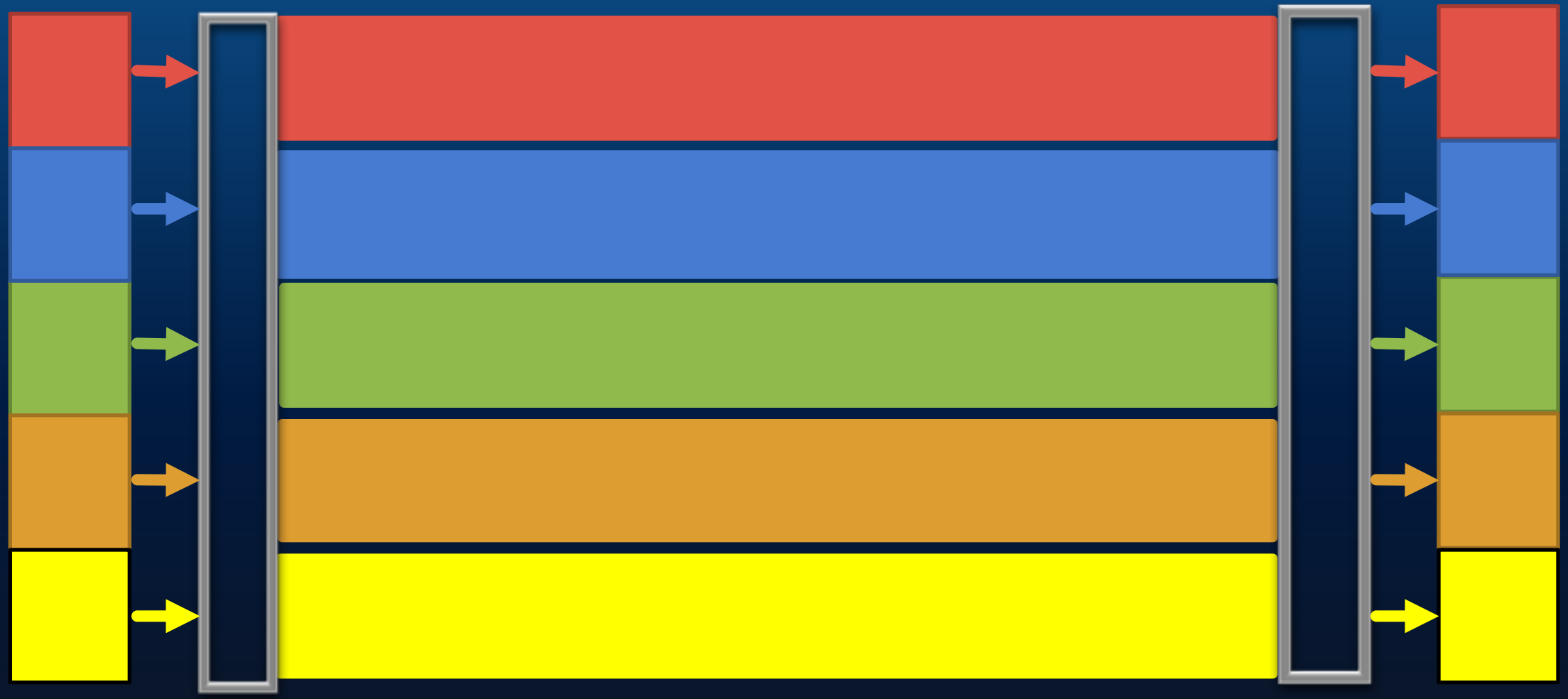
- Improve perceived performance
- Improve latency
- Single connection from client to server

- Overlap with goals and use cases
  - Easier to understand QUIC and HTTP/2 together

COMMON FEATURES

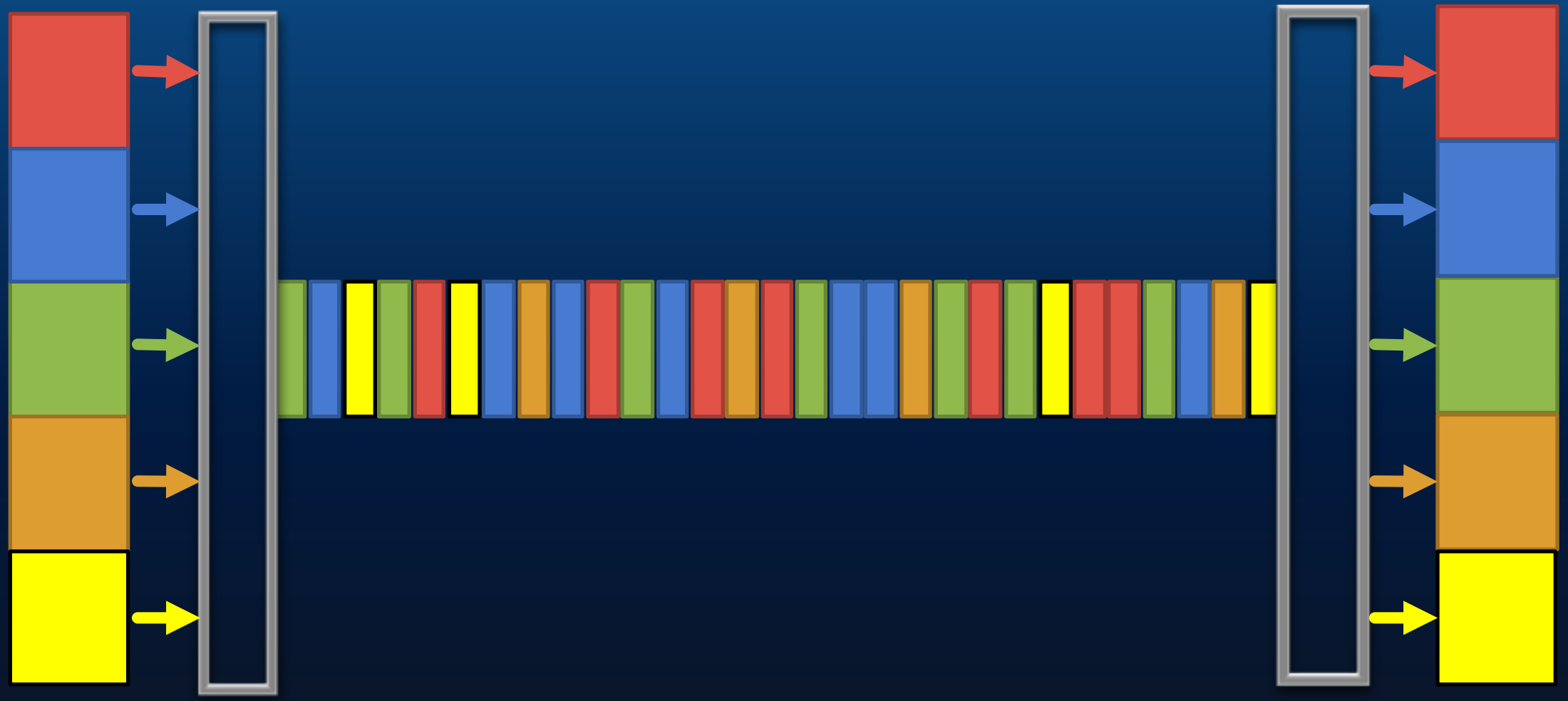
- Multiplexed Requests
- Prioritized Requests
- Compression

CURRENT



WHY USE MULTIPLE CONNECTIONS?

MULTIPLEXING

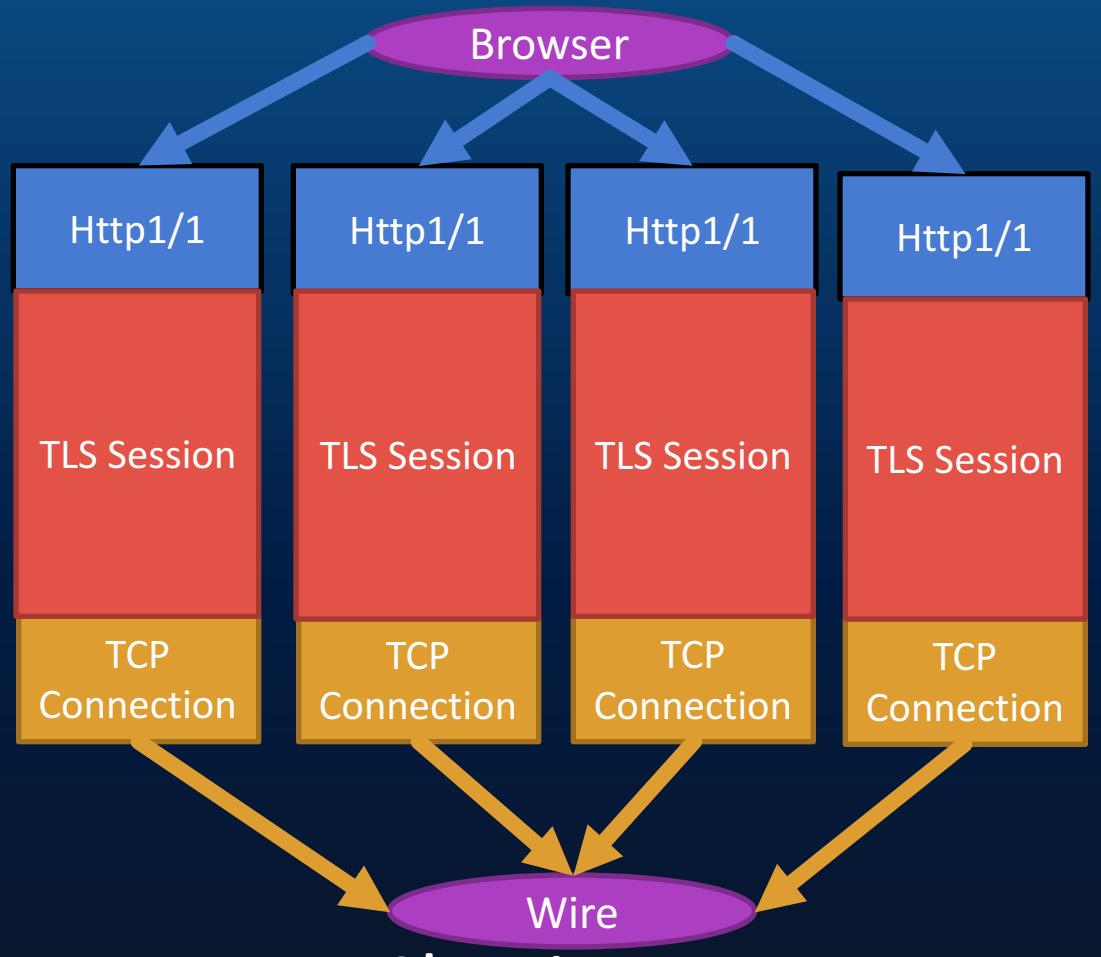


DATA FLOWS

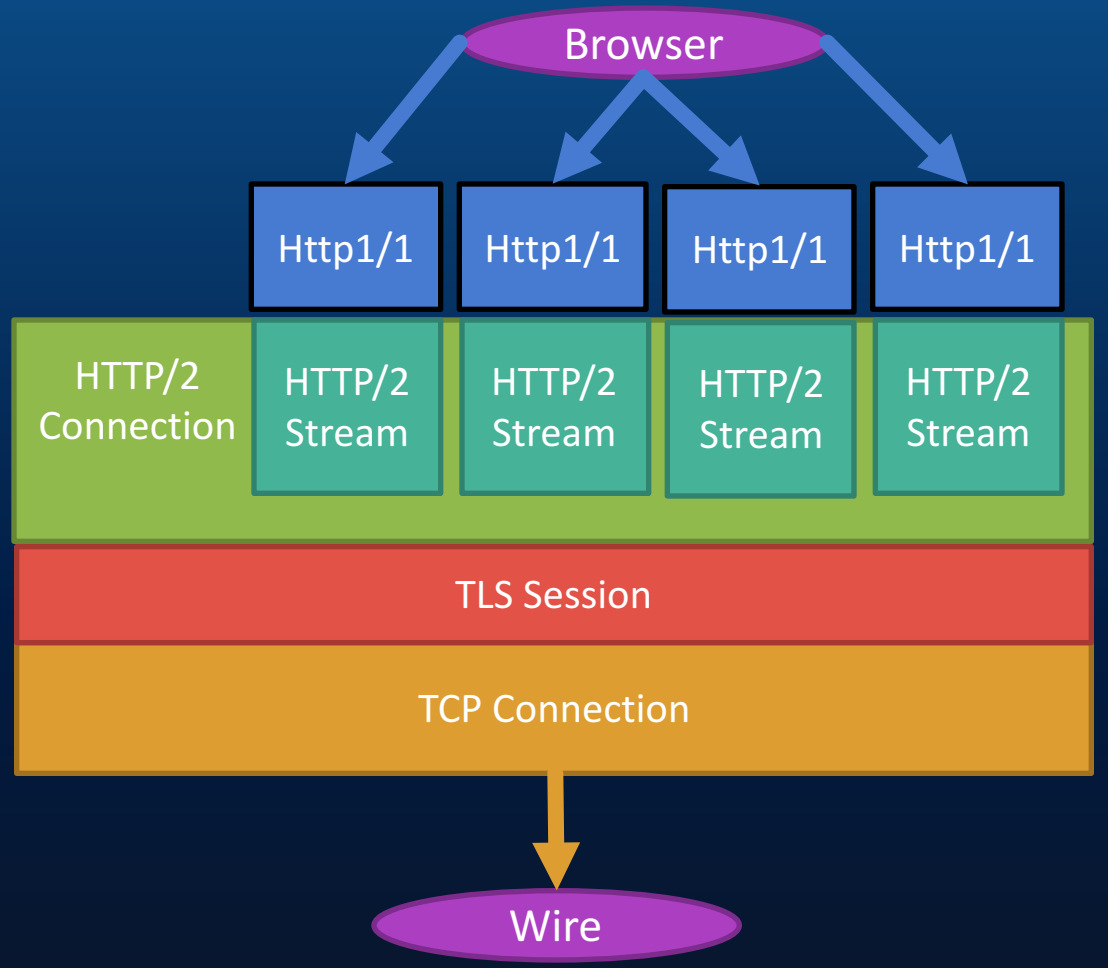
A *Single* Connection

Contains *N* Streams

TRANSPORT: HTTP VS HTTP/2

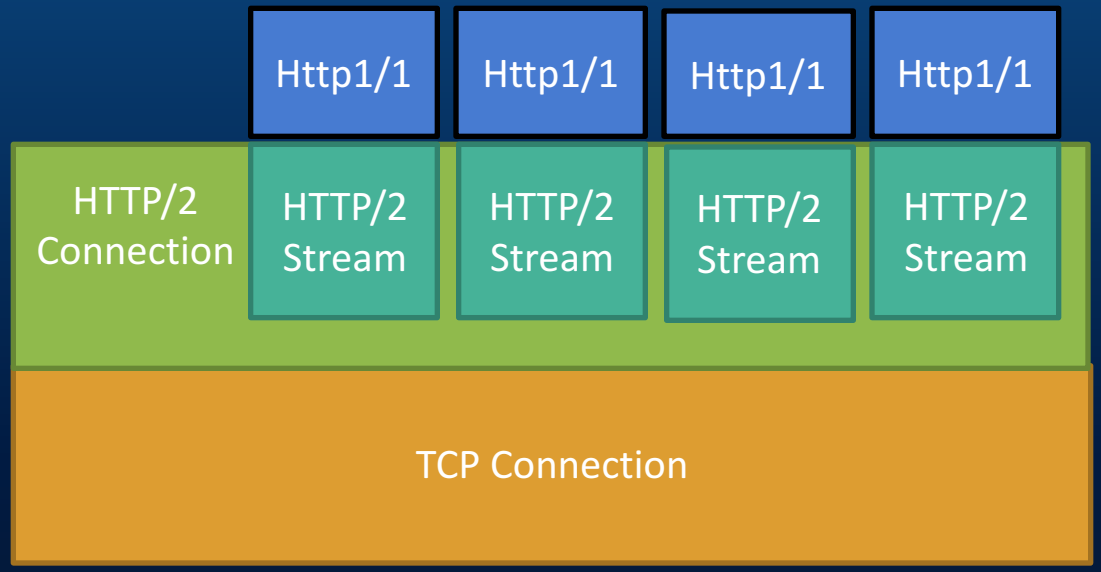


Classic HTTP

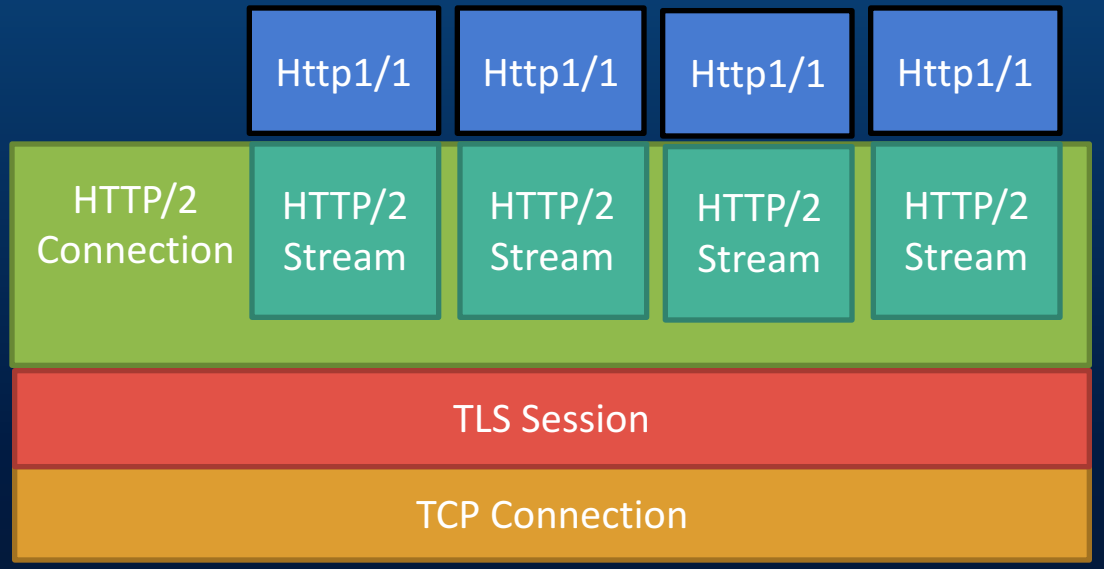


H2

CONCEPTUALLY

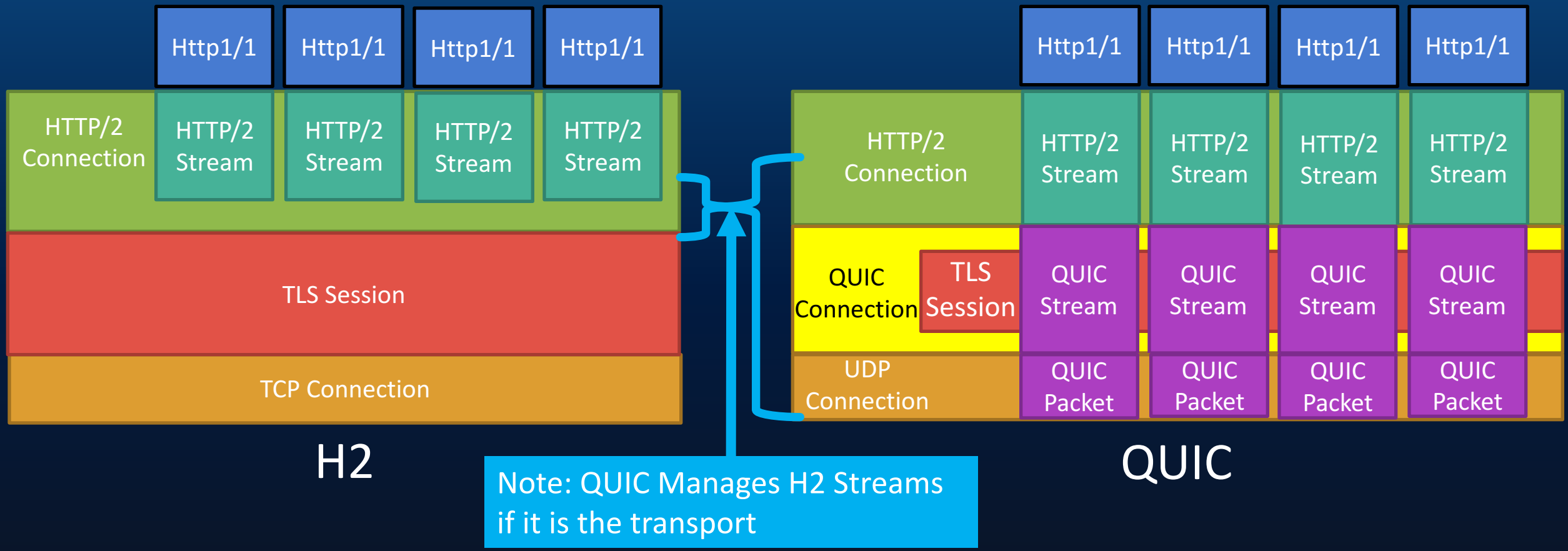


H2C



H2

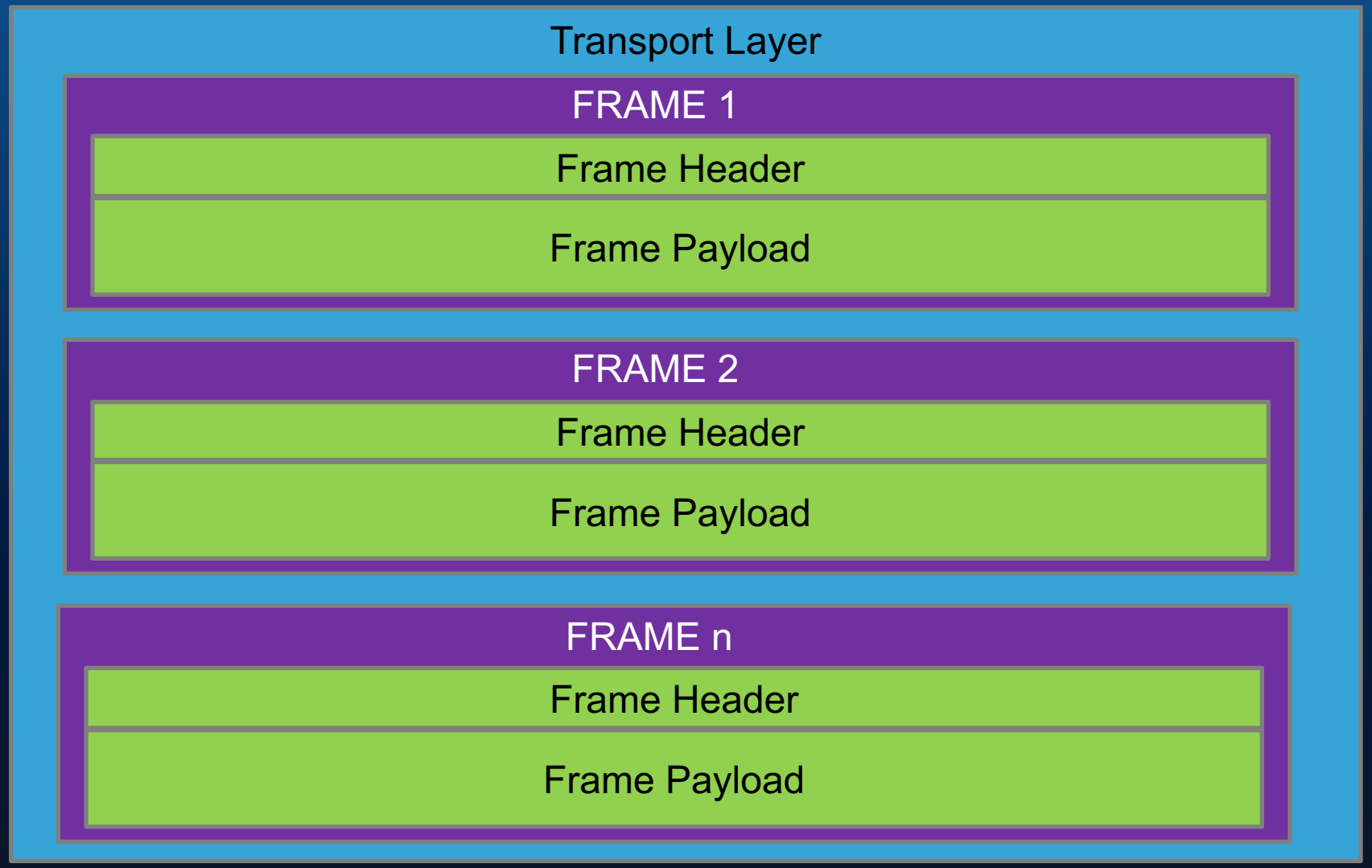
TRANSPORT: HTTP/2 VS QUIC



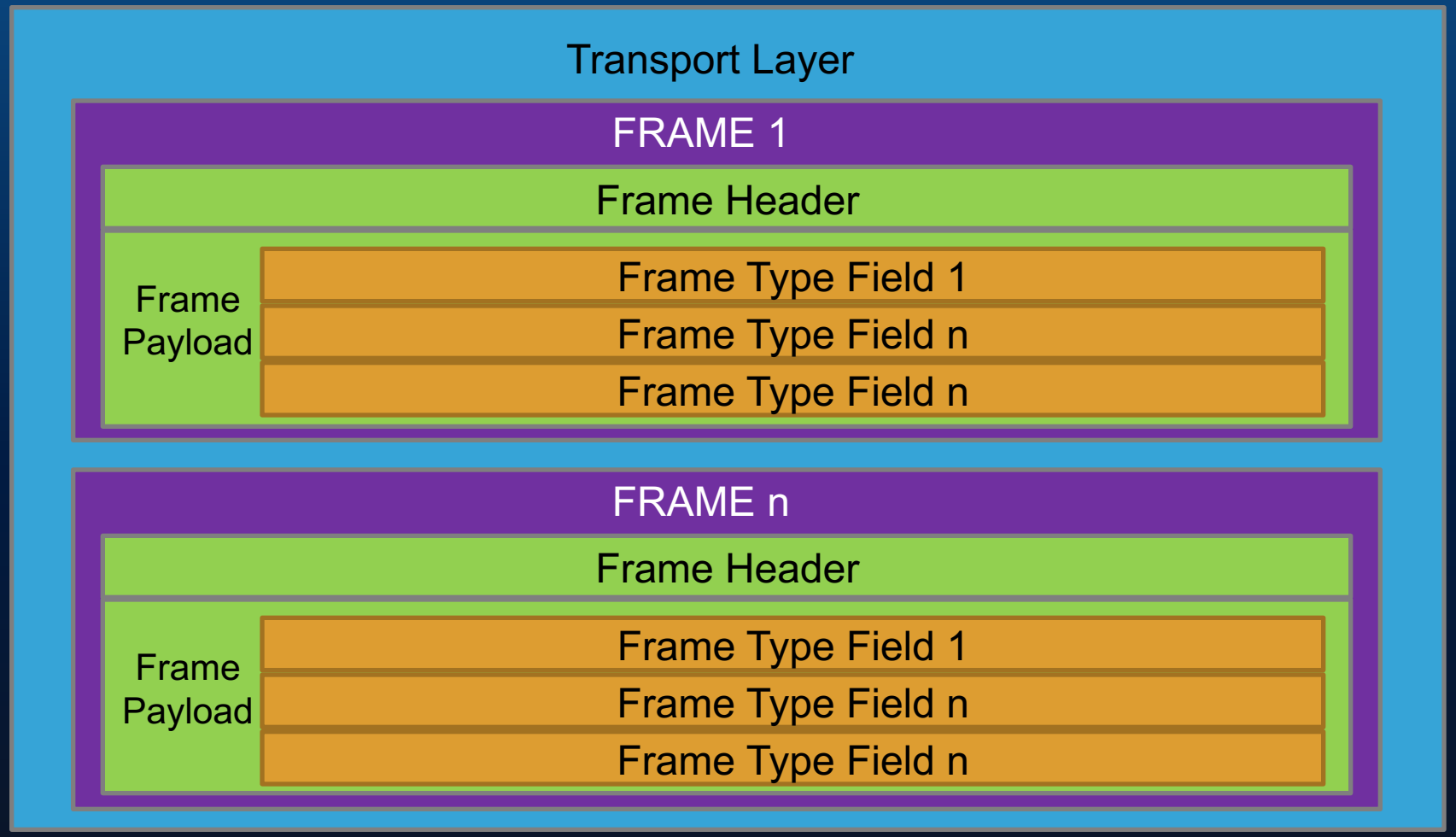
ABOUT – APPLICATION PROTOCOLS

- HTTP
  - ~20 years old
  - Uniplex
  - Text Based
  - Runs over TCP
- HTTP/2
  - Transport encapsulates HTTP to add:
    - Binary Framing
    - Multiplexed Requests
    - Prioritized Requests
    - Compression
    - Server Pushed Streams

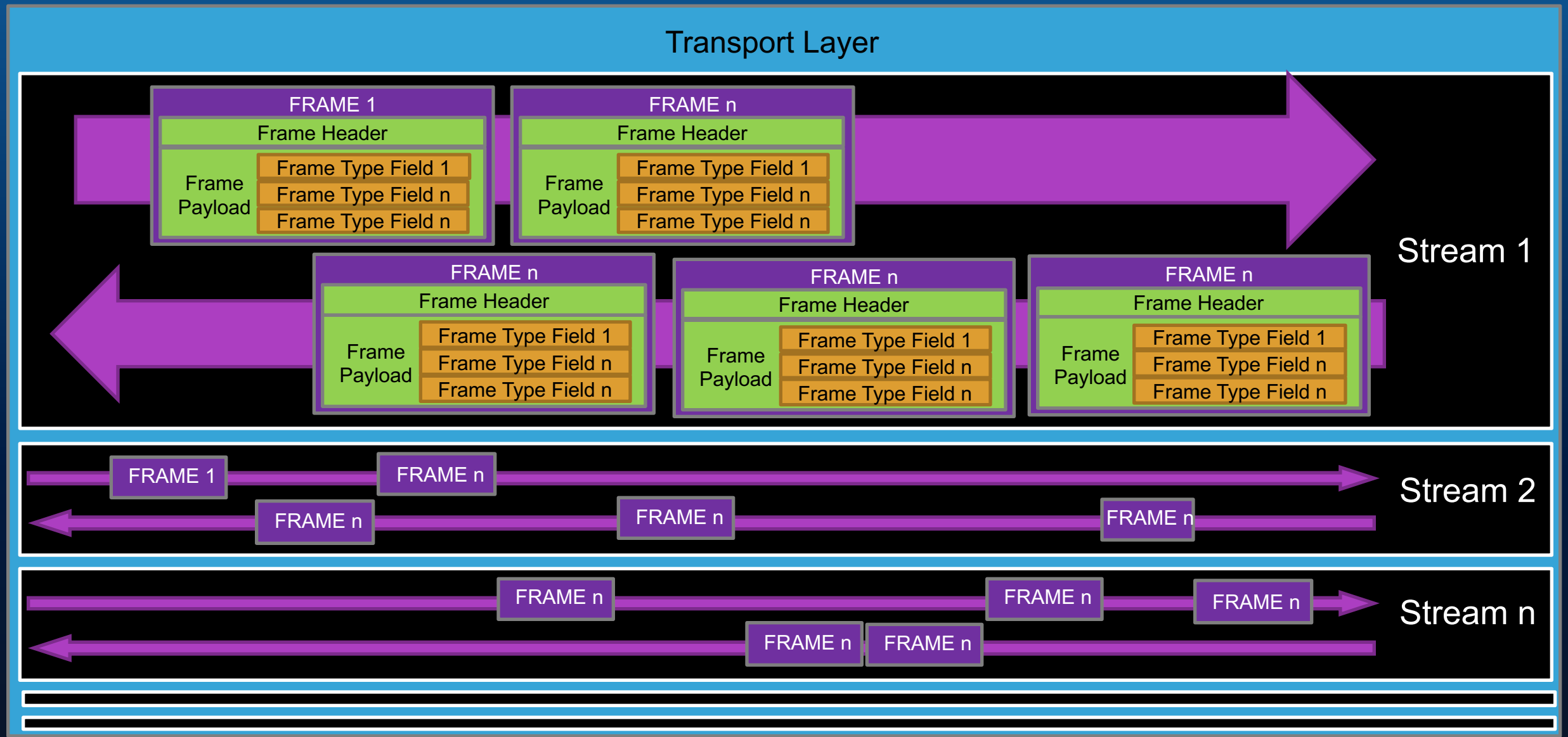
H2 STRUCTURE



H2 STRUCTURE



H2 STRUCTURE



ABOUT – HTTP/2

- Http2 composed of:
  - One connection per origin with a number of bidirectional, binary framed, streams per connection
  - Each stream has an identifier – 31-bit unsigned int, ALWAYS incrementing, never reused, odd for client initiated, even for server initiated
  - “message” analogous to HTTP request/response, composed of a sequence of frames

See https://hpbn.co/http2/

HTTP/2 CONNECTION SETUP

- Connection Establishment
  - Upgrade
    - Upgraded connections treat the first HTTP 1.1 as stream id 0x01, and switch to H2 framing once it is done…
  - Alt-svc
  - ALPN
    - H2, H2c => H2 over TLS and H2 clear-text respectively
- Note:
  - TLS with NPN <= Not supported, replaced by ALPN

HTTP/2 CONNECTION SETUP

- Prior Knowledge (Client-> Server):
  - “The client connection preface starts with a sequence of 24 octets, which in hex notation is:
    0x505249202a20485454502f322e300d0a0d0a534d0d0a0d0a

    That is, the connection preface starts with the string "PRI * HTTP/2.0\r\n\r\nSM\r\n\r\n“
    https://tools.ietf.org/html/rfc7540#section-3.5

HTTP/2 CONNECTION SETUP

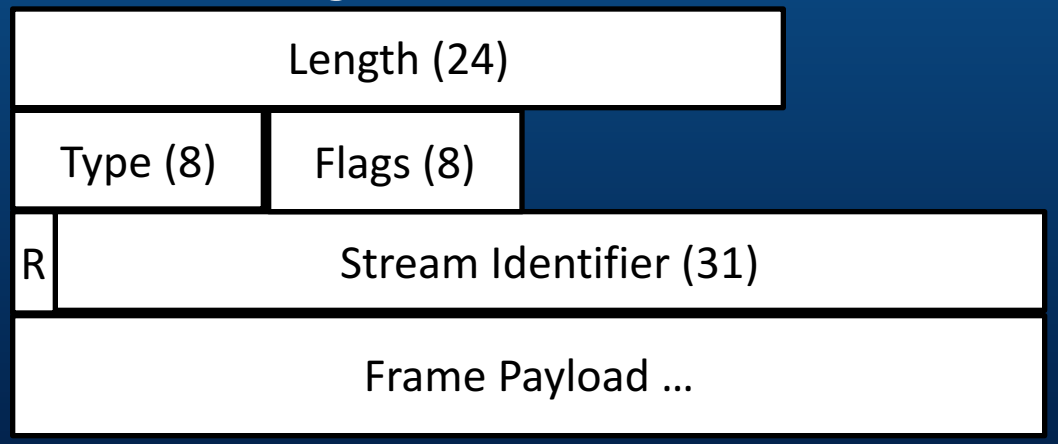
- No Prior Knowledge:
  - (http) Upgrade Header in client request (with a base64 SETTINGS payload), responds with an HTTP 101 “switching protocols”

    HTTP/1.1 101 Switching Protocols
    Connection: Upgrade
    Upgrade: h2c”
  - (https) TLS with ALPN h2, or upgrade header with h2
- Note:

  H2, H2c => h2 over tls and h2 cleartext respectively
  TLS with NPN <= Not supported, replaced by ALPN
  Upgraded connections treat the first http 1.1 as stream id 0x01, and switch to H2 framing once it is done…

H2 FRAMES

- Fixed-length header



- Variable Length Content

- Type defined by an 8-bit type code.

  Current Types:

  - DATA [Data+Padding]
  - HEADERS
  - PRIORITY
  - RST_STREAM
  - SETTINGS
  - PUSH_PROMISE
  - PING
  - GOAWAY
  - WINDOW_UPDATE
  - CONTINUATION

ABOUT – HTTP/2

Header Compression

- Compressed with HPACK (Huffman encoding), using:
  - A static table of common entries
  - A dynamic table of other items

ABOUT – HTTP/2

HTTP/2 Pitfalls?

- Connection reuse

“ Connections that are made to an origin server, either directly or through a tunnel created using the CONNECT method (Section 8.3), MAY be reused for requests with multiple different URI authority components.”

- Server push

ABOUT – QUIC

- Takes the things from HTTP/2 and adds the network layer as well
- QUIC Connections combine encryption and connection handshakes

QUIC
(QUICK UDP INTERNET CONNECTIONS)

UDP transport protocol

- *Google championed successor to SPDY*
- *Latency optimized*
- *Reliable, multiplexed*
- *Always encrypted*

Open Source

User Space

- *No OS requirements*
- *Fast-evolving*

ABOUT – QUIC

QUIC Also Adds:

- 0-RTT
- Padding
- FEC (currently disabled)
- Multipath (proposed in future)

QUIC DATA FLOWS

- ONE QUIC Connection

Contains

- N Streams

- ONE QUIC Packet

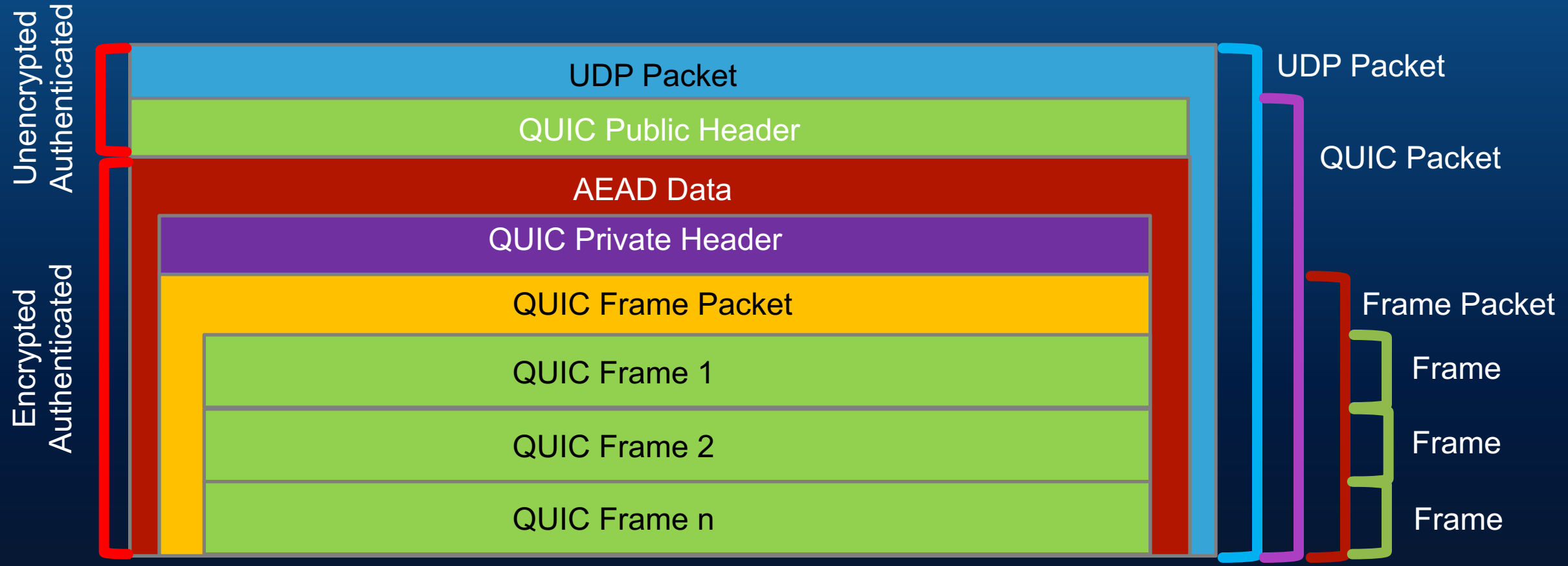
Contains

- 0-1 Frame Packets

Each containing

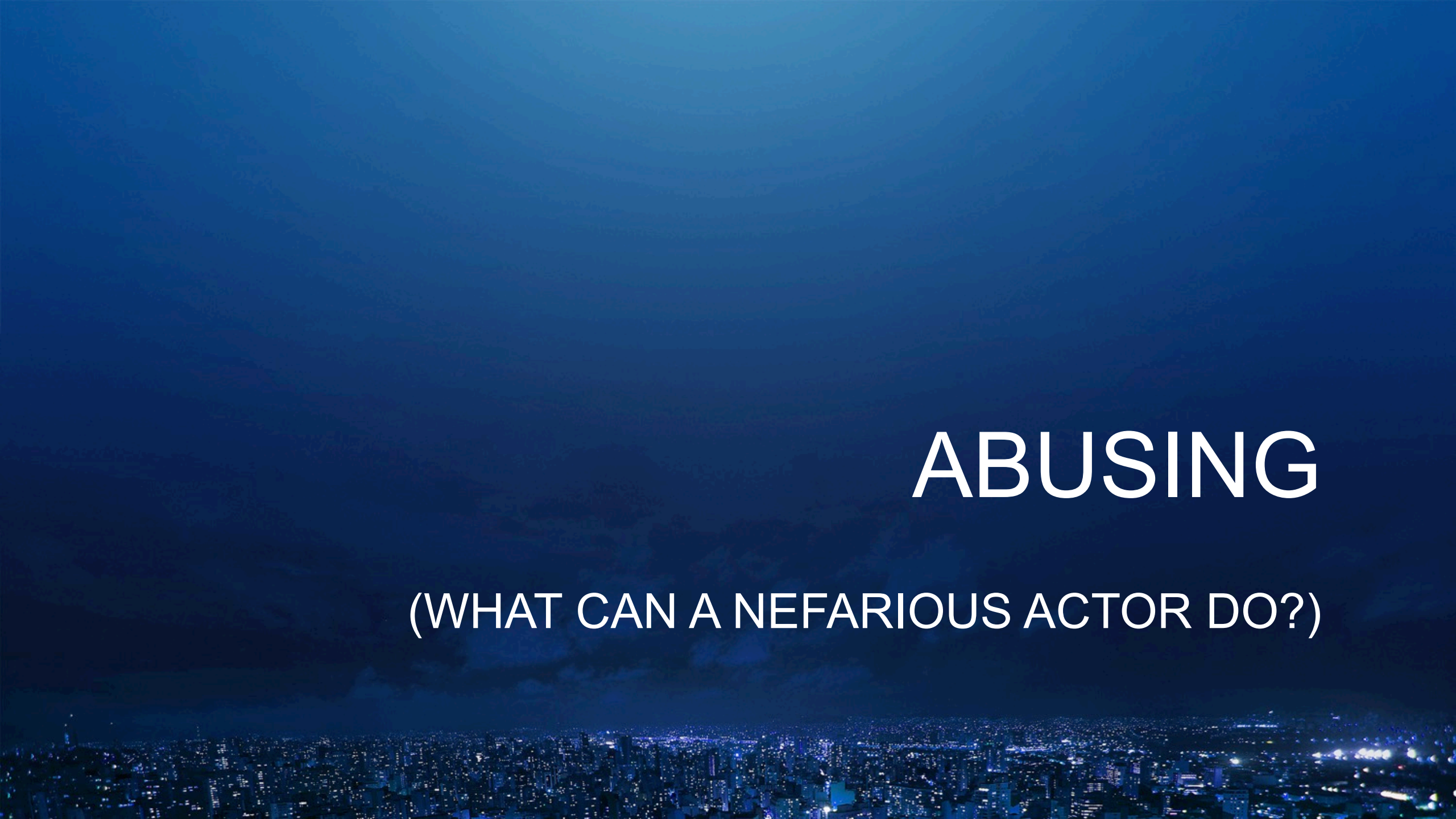
- N frames

QUIC PACKET STRUCTURE



QUIC SETUP (BROWSER)

- HTTP Header Advertisements
- Alt-svc:
  - RFC 7838
  - alt-svc quic="www.google.com:443"; p="1"; ma=600,quic=":443"; p="1"; ma=600
- Alternate-protocol
  - Old/deprecated


ABUSING
(WHAT CAN A NEFARIOUS ACTOR DO?)

SO WHY ARE THESE INTERESTING OR DANGEROUS?

- Http/2:
  - Always encrypted
  - Binary framing
  - Compression
    - Must parse to analyze
  - Much more complex state
  - Many side channels
- QUIC:
  - Encrypted, verified back to previous connections
  - User space
    - Doesn’t require a socket
  - Difficult to fingerprint
  - VERY few tools available
  - More reliable than TCP over UDP

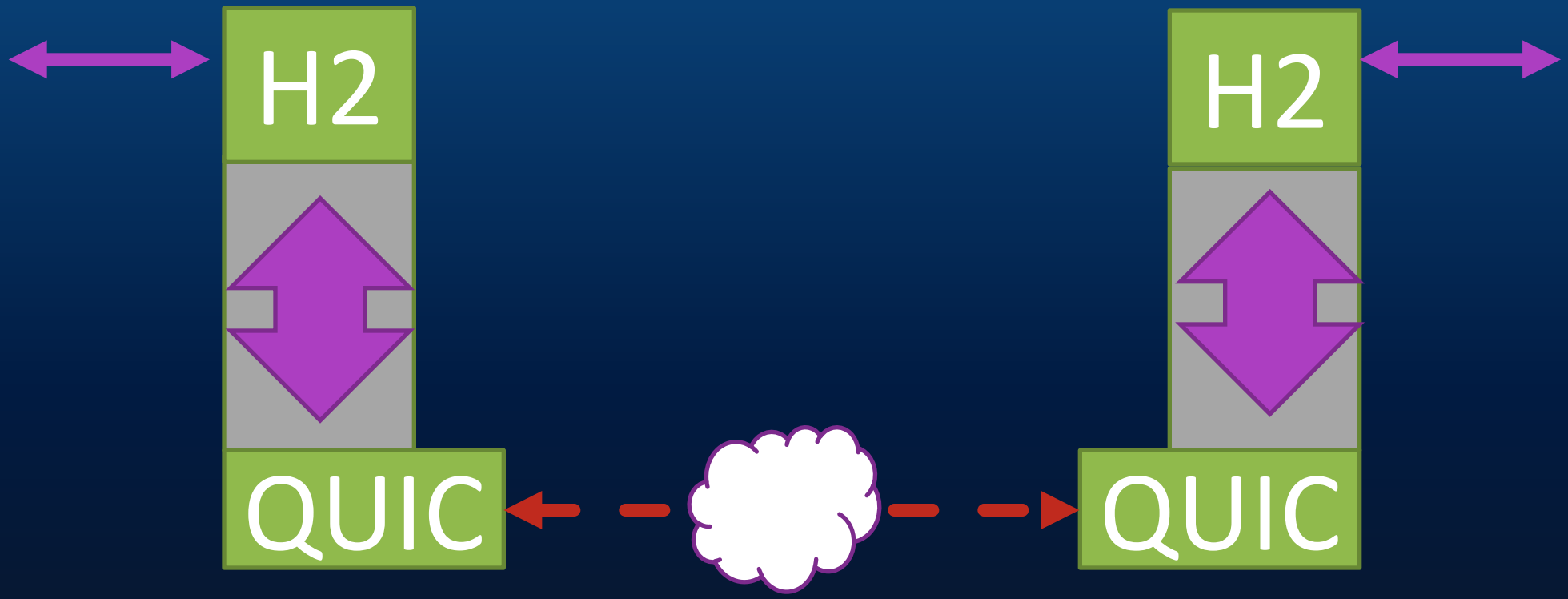
ABUSING – THE OBVIOUS

- Implementation flaws
  - Binary framing
  - Often implemented in unmanaged code
  - …
- Protocol ambiguities
  - MANY implementations
  - Fast-evolving
  - Scattered documentation

ABUSING – NEW PROTOCOLS BYPASS MONITORS

- IDS / Proxies

DEMO 1



ABUSING – NEW PROTOCOLS AND OLD TOOLS

[Quick Aside – GoLang payload injection tool by Vyrus used in these demos]

DEMO 2

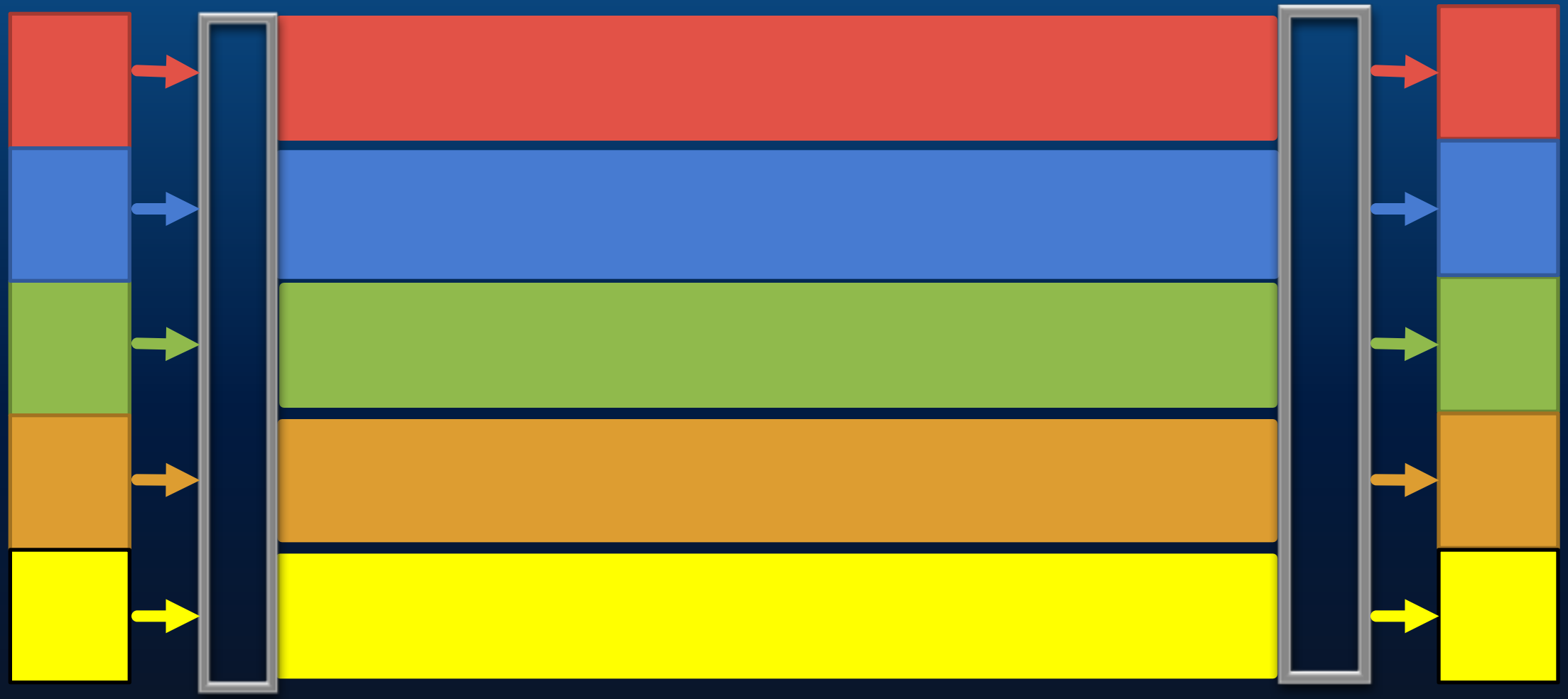
ABUSING – NEW PROTOCOLS, NEW ATTACKS

- Easy: Port-based QUIC Masquerading
- Simple: Side channels and Scrambling
- Moderate: Protocol-Embedded stego (e.g. DNS TXT field)
- Complex: Polyglots
- Extreme: Steganographic Polyglots
- Insane: Steganographic Multiplexed Polyglots

ABUSING – NEW PROTOCOLS, NEW ATTACKS

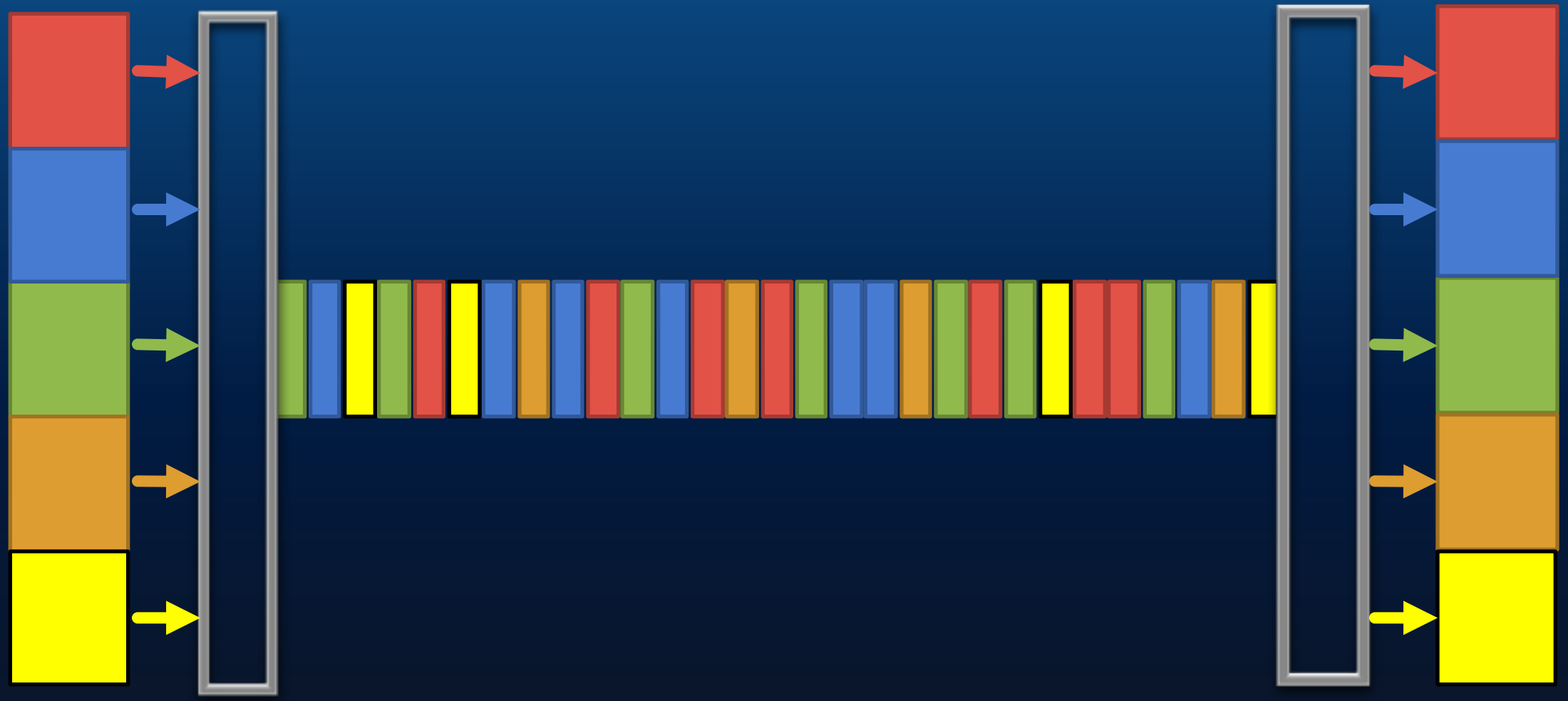
- Fragmentation & agility
  - Multi-connection
  - Multi-path
  - Multi-stream

CURRENT

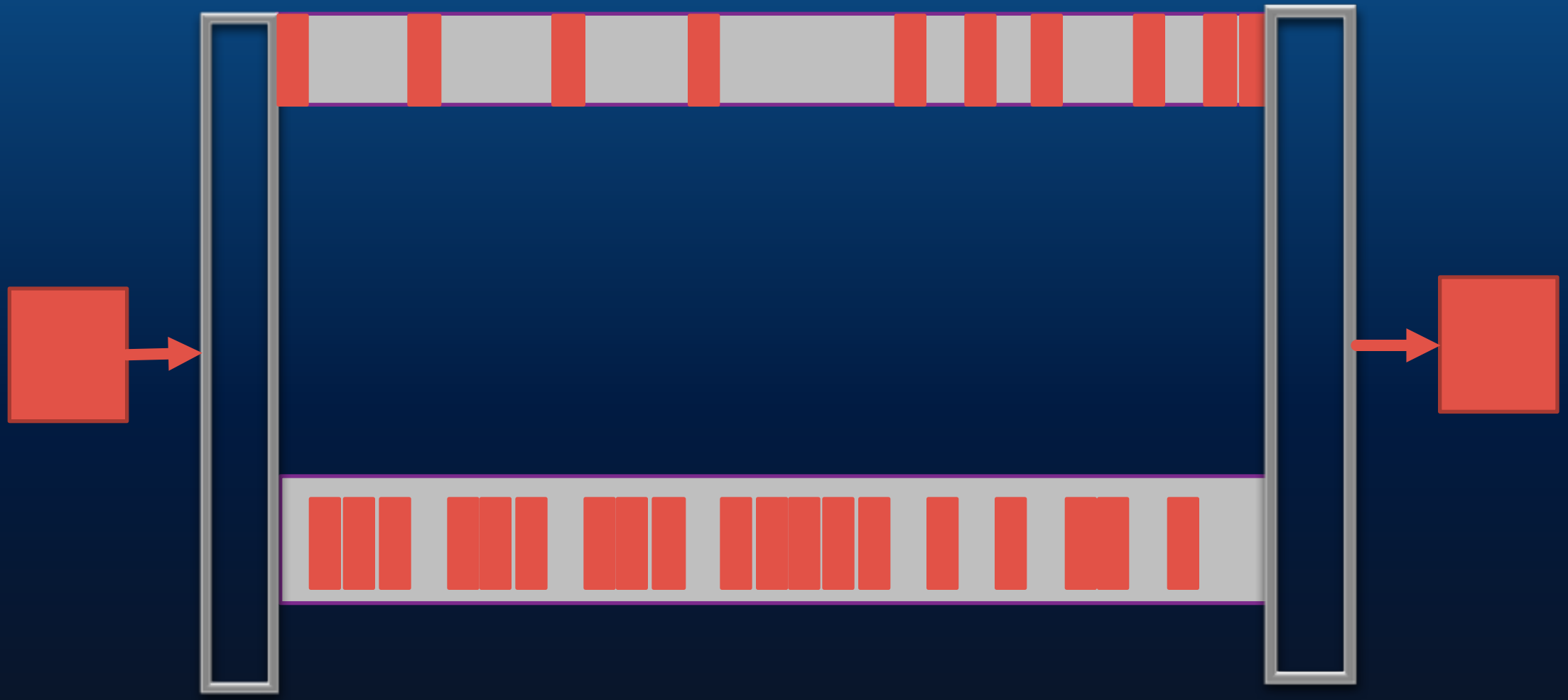


WHY USE MULTIPLE CONNECTIONS?

MULTIPLEXING



MULTIPATH / MULTICONNECTION



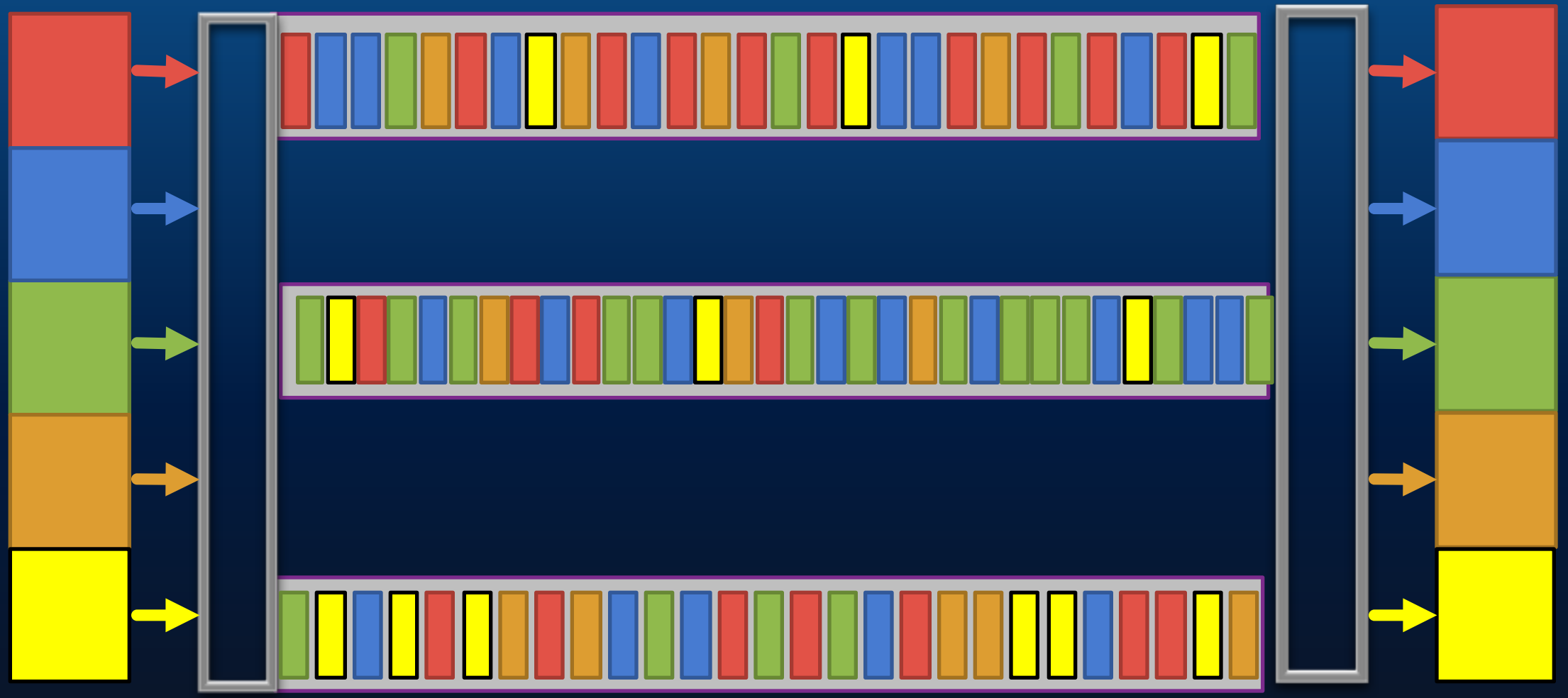
ABUSING – NEW PROTOCOLS, NEW ATTACKS

- And if we combine multiplexing and multiconnection/path…

ABUSING – NEW PROTOCOLS, NEW ATTACKS

- Fragmentation & agility
  - Multi-connection
  - Multi-path
  - Multi-stream

MULTIPATH MULTIPLEXED



ABUSING – NEW PROTOCOLS, NEW ATTACKS

- Cross-path fragmentation
- Cross-path agility
- Multi-stream fragmentation
- Multi-stream agility

ABUSING – NEW PROTOCOLS, NEW ATTACKS

- Forward Error Correction
  - **REMOVED AT PRESENT - https://groups.google.com/a/chromium.org/d/msg/proto-quic/Z5qKkk2XZe0/yzAqOgNWHgAJ https://docs.google.com/document/d/1Hg1SaLEl6T4rEU9j-isovCo8VEjjnuCPTcLNJewj7Nk/edit**
  - Fake Packet Injection (False Checksums)
  - Dropping/Corrupting packets

ANALYZING & DEFENDING

(WHAT DO WE DO WHEN WE SEE THESE THINGS)

ANALYZING & DEFENDING – DETECT CLIENT TRAFFIC

- HTTP/2 Client
  - ALPN
  - Upgrade headers
- QUIC Client Traffic
  - UDP Ports 80 and 443
  - Bidirectional patterns of communications
  - No static identifier in header, you have to parse it
- QUIC Detector

DEMO 3

ANALYZING & DEFENDING – DETECT SERVERS

- HTTP/2 Server
  - ALPN
  - Upgrade Headers
- QUIC Server Traffic
  - UDP Ports 80 and 443
  - QUIC Scanner…

DEMO 4

ANALYZING & DEFENDING - BLOCK

- H2
  - Transparent proxies
    - Don’t support HTTP2 outbound
    - Rewrite or remove upgrade headers
  - HTTPS ALPN
  - HTTP/H2 on nonstandard ports (80, 443, 8080, 8443)

- QUIC
  - UDP Ports 80 and 443
  - Application/policy Settings (Chrome)
  - Fingerprinted/detected/parsed QUIC

ANALYZING & DEFENDING - ANALYSE

- H2
  - Wireshark
  - Chrome
  - H2i
  - Nghttp
  - curl
- QUIC
  - Wireshark
  - Chrome

ANALYZING HTTP/2 IN WIRESHARK

Use an SSLKEYLOGFILE

The dissector's pretty good

ANALYZING HTTP/2 IN WIRESHARK

Use an SSLKEYLOGFILE

The dissector's pretty good

No.	Time	Source	Destination	Protocol	Length	Stream Identifier
86	3.930832433	192.168.1.197	179.60.193.36	HTTP2	151	17
87	3.930870174	192.168.1.197	179.60.193.36	HTTP2	151	19
88	3.930906922	192.168.1.197	179.60.193.36	HTTP2	155	21
95	3.959659717	179.60.193.36	192.168.1.197	HTTP2	1464	1
96	3.959854307	179.60.193.36	192.168.1.197	HTTP2	1464	1,1
98	3.959869355	179.60.193.36	192.168.1.197	HTTP2	1464	1
101	3.959898025	179.60.193.36	192.168.1.197	TLSv1.2	1464	1
105	3.959966438	179.60.193.36	192.168.1.197	HTTP2	1227	1
115	3.974450030	179.60.193.36	192.168.1.197	HTTP2	108	3
116	3.974460780	179.60.193.36	192.168.1.197	HTTP2	108	5
117	3.974462652	179.60.193.36	192.168.1.197	HTTP2	108	7
120	4.003507969	179.60.193.36	192.168.1.197	HTTP2	1464	9
121	4.003543031	179.60.193.36	192.168.1.197	HTTP2	1464	11,3
130	4.003705564	179.60.193.36	192.168.1.197	HTTP2	1464	3
132	4.003711897	179.60.193.36	192.168.1.197	HTTP2	1464	7
133	4.003713289	179.60.193.36	192.168.1.197	HTTP2	1464	7
135	4.003817896	179.60.193.36	192.168.1.197	HTTP2	1464	13,11,5
151	4.017766949	179.60.193.36	192.168.1.197	HTTP2	1464	5
154	4.047566888	179.60.193.36	192.168.1.197	HTTP2	1464	5
159	4.047590257	179.60.193.36	192.168.1.197	HTTP2	1464	11,15
160	4.047591650	179.60.193.36	192.168.1.197	HTTP2	1464	17,9,13,9
165	4.047598965	179.60.193.36	192.168.1.197	HTTP2	1464	13,19,21
166	4.047826021	179.60.193.36	192.168.1.197	HTTP2	1464	17
168	4.047833359	179.60.193.36	192.168.1.197	HTTP2	1464	17
169	4.047835114	179.60.193.36	192.168.1.197	HTTP2	1464	15,21
187	4.048098068	179.60.193.36	192.168.1.197	HTTP2	1464	15
193	4.061501607	179.60.193.36	192.168.1.197	HTTP2	1464	15
211	4.091981695	179.60.193.36	192.168.1.197	HTTP2	1464	21
225	4.092186260	179.60.193.36	192.168.1.197	HTTP2	1464	21,19,19
235	4.092500279	179.60.193.36	192.168.1.197	HTTP2	1464	1
237	4.092503968	179.60.193.36	192.168.1.197	HTTP2	1464	1
238	4.092551621	179.60.193.36	192.168.1.197	HTTP2	1464	1

ANALYZING HTTP/2 IN CHROME

chrome://net-internals/#http2

HTTP/2 (capturing events (5:59))

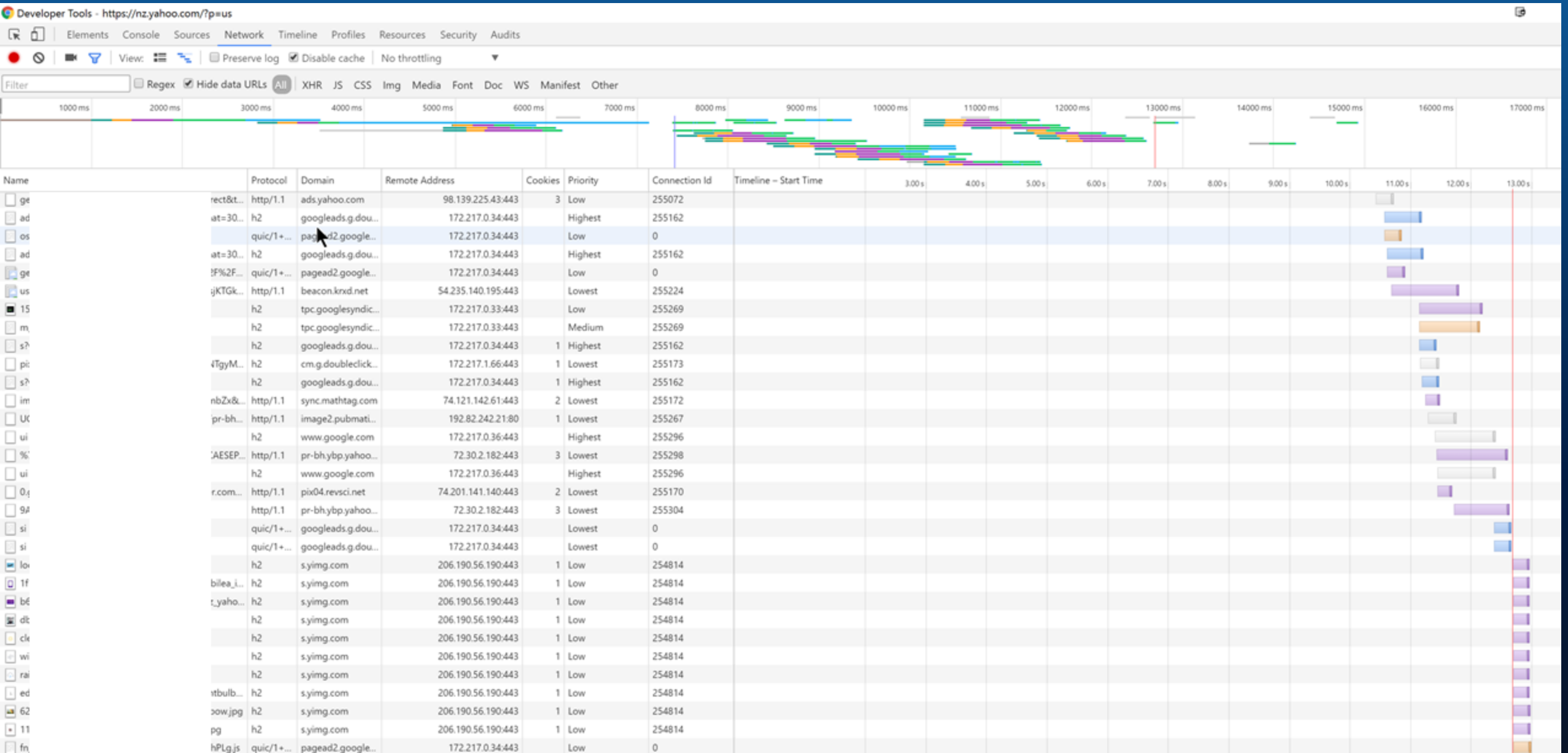
- HTTP/2 Enabled: true
- SPDY/3.1 Enabled: false
- Use Alternative Service: true
- ALPN Protocols: h2,http/1.1
- NPN Protocols: undefined

HTTP/2 sessions

View live HTTP/2 sessions

Host	Proxy	ID	Protocol Negotiated	Active streams	Unclaimed pushed	Max	Initiated	Pushed	Pushed and claimed	Abandoned	Received frames	Secure	Sent settings	Received settings	Send window	Receive window	Unacked received data	Error
cm.dpclk.com:443	direct://	41853	h2	0	0	250	1	0	0	0	1	true	true	true	65535	15728640	0	0
play.google.com:443	direct://	253855	h2	0	0	100	1	0	0	0	2	true	true	true	1048158	15728640	153	0
plus.google.com:443	direct://	253306	h2	0	0	100	1	0	0	0	3	true	true	true	1048467	15728640	408	0
twitter.com:443	direct://	229415	h2	0	0	100	1777	0	0	0	3989	true	true	true	64983	15728640	3550535	0
www.google.co.nz:443	direct://	253540	h2	0	0	100	1	0	0	0	2	true	true	true	1048576	15728640	365	0
clients4.google.com:443	direct://	253919	h2	0	0	100	0	0	0	0	0	true	true	true	1048576	15728640	0	0
play.google.com:443	direct://	253845	h2	0	0	100	1	0	0	0	2	true	true	true	1048576	15728640	0	0

ANALYZING HTTP/2 IN CHROME Dev tools



ANALYZING HTTP/2 IN CHROME Dev tools

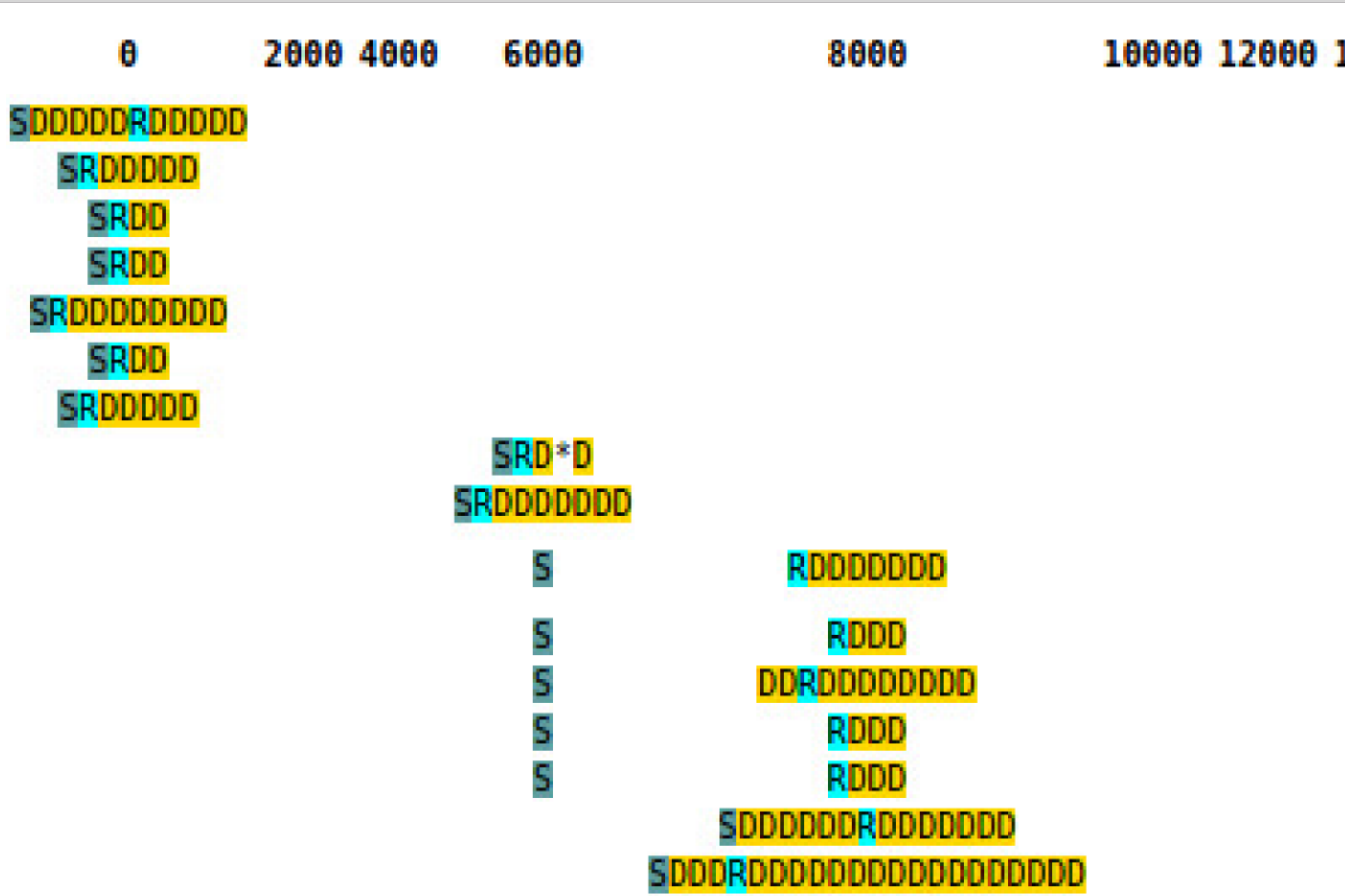
Name		Protocol	Domain	Remote Address	Cookies	Priority	Connection Id
ge	rect&t...	http/1.1	ads.yahoo.com	98.139.225.43:443	3	Low	255072
ad	at=30...	h2	googleads.g.dou...	172.217.0.34:443		Highest	255162
os		quic/1+...	pagead2.google...	172.217.0.34:443		Low	0
ad	at=30...	h2	googleads.g.dou...	172.217.0.34:443		Highest	255162
ge	!F%2F...	quic/1+...	pagead2.google...	172.217.0.34:443		Low	0
us	ijKTGk...	http/1.1	beacon.krxd.net	54.235.140.195:443		Lowest	255224
15		h2	tpc.googlesyndic...	172.217.0.33:443		Low	255269
m		h2	tpc.googlesyndic...	172.217.0.33:443		Medium	255269
s?		h2	googleads.g.dou...	172.217.0.34:443	1	Highest	255162
pi	4TgyM...	h2	cm.g.doubleclick...	172.217.1.66:443	1	Lowest	255173
s?		h2	googleads.g.dou...	172.217.0.34:443	1	Highest	255162
im	nbZx&...	http/1.1	sync.mathtag.com	74.121.142.61:443	2	Lowest	255172
UC	pr-bh...	http/1.1	image2.pubmati...	192.82.242.21:80	1	Lowest	255267

ANALYZING HTTP/2 IN CHROME-HTTP2-LOG-PARSER

file:///home/username/Desktop/chromeLogs/output.html

S HTTP2_SESSION_SEND_HEADERS
R HTTP2_SESSION_RECV_HEADERS
D HTTP2_SESSION_RECV_DATA
P HTTP2_SESSION_RECV_PUSH_PROMISE
A HTTP2_STREAM_ADOPTED_PUSH_STREAM
* OTHER

Stream ID	Path
25	/
27	/portal/wikipedia.org/assets/j
29	/portal/wikipedia.org/assets/j
31	/portal/wikipedia.org/assets/i
33	/portal/wikipedia.org/assets/i
35	/portal/wikipedia.org/assets/i
37	/portal/wikipedia.org/assets/i
39	/



ANALYZING HTTP/2 IN H2I

```
username@bhubu ~/Desktop $ h2i www.facebook.com
Connecting to www.facebook.com:443 ...
Connected to 179.60.193.36:443
Negotiated protocol "h2"
Sending: []
[FrameHeader SETTINGS len=30]
  [HEADER_TABLE_SIZE = 4096]
  [MAX_FRAME_SIZE = 16384]
  [MAX_HEADER_LIST_SIZE = 131072]
  [MAX_CONCURRENT_STREAMS = 100]
  [INITIAL_WINDOW_SIZE = 65536]
[FrameHeader WINDOW_UPDATE len=4]
  Window-Increment = 65537
[FrameHeader SETTINGS flags=ACK len=0]
h2i> headers
(as HTTP/1.1)> GET / HTTP/1.0
(as HTTP/1.1)>
Opening Stream-ID 1:
 :authority =
 :method = GET
 :path = /
 :scheme = https
[FrameHeader WINDOW_UPDATE stream=1 len=4]
  Window-Increment = 10420224

[FrameHeader HEADERS flags=END_HEADERS stream=1 len=144]
  :status = "301"
  location = "https://www.facebook.com/"
  content-type = "text/html"
  x-fb-debug = "aQa2pHbnyIpvW4Xkv5d668s4y2QTzH7nZJYBYl0MES
XBpOZRBesmZlkTak7AS9TQ=="
  date = "Thu, 21 Jul 2016 04:54:20 GMT"
  content-length = "0"
[FrameHeader DATA flags=END_STREAM stream=1 len=0]
```

ANALYZING HTTP/2 IN NGH

```
username@bhubu ~/Desktop $ nghttp -nvas https://www.cloudflare.com
[  0.023] Connected
[  0.047][NPN] server offers:
          * h2
          * spdy/3.1
          * http/1.1
The negotiated protocol: h2
[  0.068] recv SETTINGS frame <length=18, flags=0x00, stream_id=0>
          (niv=3)
          [SETTINGS_MAX_CONCURRENT_STREAMS(0x03):128]
          [SETTINGS_INITIAL_WINDOW_SIZE(0x04):65536]
          [SETTINGS_MAX_FRAME_SIZE(0x05):16777215]
[  0.068] recv WINDOW_UPDATE frame <length=4, flags=0x00, stream_id=0>
          (window_size_increment=2147418112)
[  0.068] send SETTINGS frame <length=12, flags=0x00, stream_id=0>
          (niv=2)
          [SETTINGS_MAX_CONCURRENT_STREAMS(0x03):100]
          [SETTINGS_INITIAL_WINDOW_SIZE(0x04):65535]
[  0.068] send SETTINGS frame <length=0, flags=0x01, stream_id=0>
          ; ACK
          (niv=0)
```

```
[  0.096] send HEADERS frame <length=32, flags=0x25, stream_id=49>
          ; END_STREAM | END_HEADERS | PRIORITY
          (padlen=0, dep_stream_id=5, weight=2, exclusive=0)
          ; Open new stream
          :method: GET
          :path: /js/index.js?v=1468968420
          :scheme: https
          :authority: www.cloudflare.com
          accept: */*
          accept-encoding: gzip, deflate
          user-agent: nghttp2/1.11.0-DEV
[  0.100] recv (stream_id=2) :status: 200
[  0.100] recv (stream_id=2) date: Thu, 21 Jul 2016 04:57:42 GMT
[  0.100] recv (stream_id=2) content-type: application/x-javascript
[  0.100] recv (stream_id=2) set-cookie: __cfduid=d1e23a2425fca2c67ad09bf406441df361
[  0.100] recv (stream_id=2) last-modified: Thu, 29 Oct 2015 20:59:13 GMT
[  0.100] recv (stream_id=2) etag: W/"563288a1-14979"
[  0.100] recv (stream_id=2) expires: Fri, 21 Jul 2017 04:57:42 GMT
[  0.100] recv (stream_id=2) cache-control: public, max-age=31536000
[  0.100] recv (stream_id=2) content-encoding: gzip
[  0.100] recv (stream_id=2) cf-cache-status: HIT
[  0.100] recv (stream_id=2) vary: Accept-Encoding
[  0.100] recv (stream_id=2) server: cloudflare-nginx
[  0.100] recv (stream_id=2) cf-ray: 2c5c12d8aeb518ea-AKL
[  0.100] recv HEADERS frame <length=358, flags=0x04, stream_id=2>
```

id	responseEnd	requestStart	process	code	size	request path
13	+27.86ms	+246us	27.61ms	200	5K	/
2	+64.09ms *	+26.90ms	37.19ms	200	29K	/js/jquery-2.1.4-min.js
25	+64.18ms	+28.09ms	36.09ms	200	128	/media/icons/icon-bolt.svg
21	+67.71ms	+28.01ms	39.70ms	200	4K	/media/cloudflare-logo.png
45	+67.83ms	+28.48ms	39.35ms	200	1K	/js/banner.js?v=1468968420
15	+72.21ms	+27.87ms	44.33ms	200	16K	/favicon.ico
31	+73.15ms	+28.21ms	44.94ms	200	151	/media/icons/icon-dns.svg
41	+73.24ms	+28.40ms	44.84ms	200	1K	/js/form.js?v=1468968420
33	+73.53ms	+28.25ms	45.28ms	200	1K	/js/core.js?v=1468968420
47	+73.59ms	+28.52ms	45.07ms	200	577	/js/global.js?v=1468968420
49	+74.23ms	+28.56ms	45.67ms	200	320	/js/index.js?v=1468968420
19	+74.30ms	+27.97ms	46.33ms	200	687	/css/home-page.css?v=1468968420
23	+74.35ms	+28.05ms	46.30ms	200	191	/media/icons/icon-pin.svg
29	+74.42ms	+28.17ms	46.25ms	200	150	/media/icons/icon-lock.svg
39	+75.13ms	+28.36ms	46.77ms	200	563	/js/validation.js?v=1468968420
17	+77.55ms	+27.92ms	49.63ms	200	8K	/css/main.css?v=1468968420
43	+77.59ms	+28.44ms	49.15ms	200	1K	/js/tooltip.js?v=1468968420
35	+77.61ms	+28.28ms	49.33ms	200	567	/js/analytics.js?v=1468968420
37	+78.03ms	+28.32ms	49.71ms	200	1K	/js/translations.js?v=1468968420
27	+78.04ms	+28.13ms	49.91ms	200	140	/media/icons/icon-shield.svg

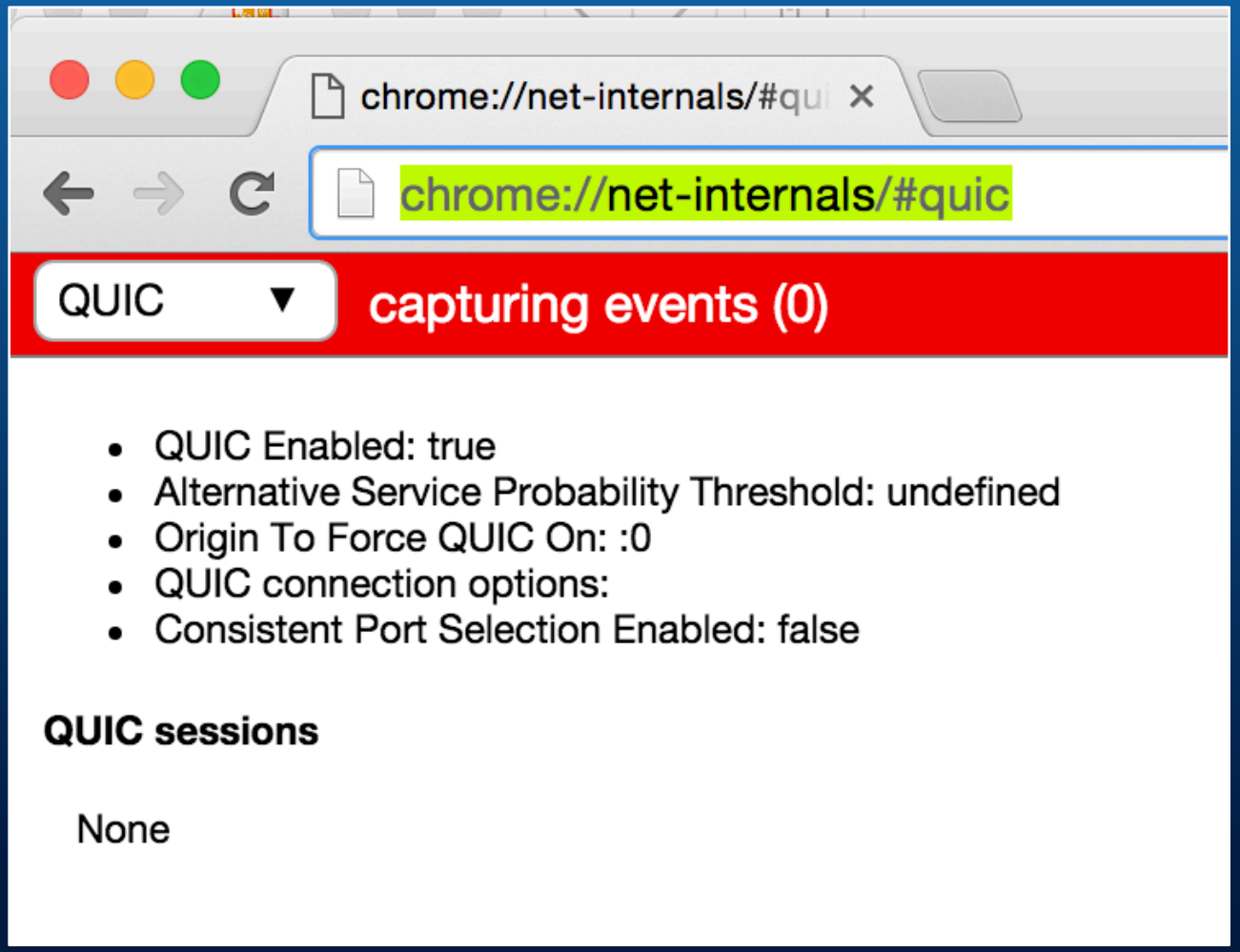
ANALYZING HTTP/2 IN CURL

```
username@bhubu ~/Desktop $ curl -vso /dev/null --http2 https://www.cloudflare.com
* Rebuilt URL to: https://www.cloudflare.com/
*   Trying 198.41.214.162...
* Connected to www.cloudflare.com (198.41.214.162) port 443 (#0)
* Cipher selection: ALL:!EXPORT:!EXPORT40:!EXPORT56:!aNULL:!LOW:!RC4:@STRENGTH
* successfully set certificate verify locations:
*   CAfile: /etc/ssl/certs/ca-certificates.crt
```

```
* Using HTTP2, server supports multi-use
* Connection state changed (HTTP/2 confirmed)
* TCP_NODELAY set
* Copying HTTP/2 data in stream buffer to connection buffer after upgrade: len=0
* Using Stream ID: 1 (easy handle 0x766b90)
> GET / HTTP/1.1
> Host: www.cloudflare.com
> User-Agent: curl/7.46.0
> Accept: */*
>
< HTTP/2.0 200
< date:Thu, 21 Jul 2016 05:03:30 GMT
< content-type:text/html
< set-cookie:__cfduid=dad932fd84c5914a313eca3807ccc605e1469077410; expires=Fri, 21
GMT; path=/; domain=.cloudflare.com; HttpOnly
< last-modified:Tue, 19 Jul 2016 22:47:05 GMT
< cf-cache-status:HIT
< expires:Thu, 21 Jul 2016 09:03:30 GMT
< cache-control:public, max-age=14400
< server:cloudflare-nginx
< cf-ray:2c5c1b57692918f6-AKL
< cf-h2-pushed:</js/jquery-2.1.4-min.js>
```

DEBUGGING QUIC

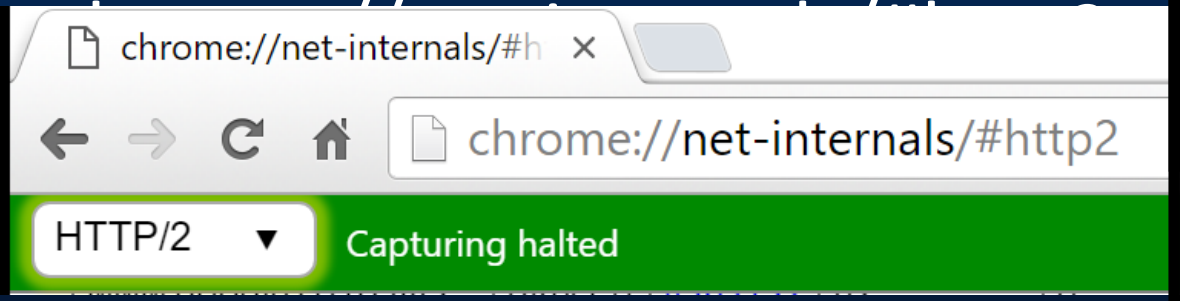
- Chrome:



- chrome://net-internals/#quic

ANALYZING QUIC

- Chrome:



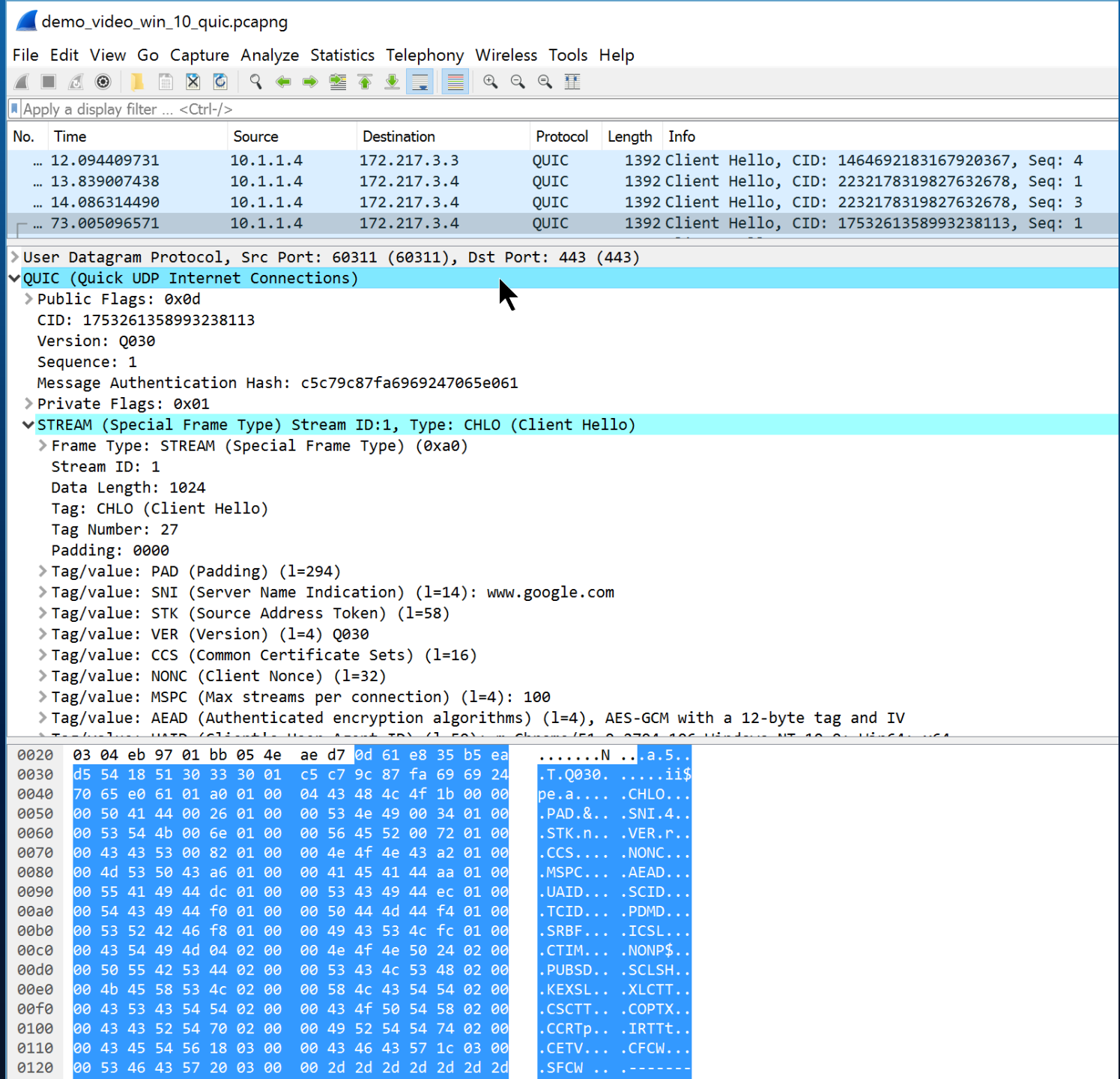
Alternative Service Mappings

Host	Alternative Service
encrypted-tbn3.gstatic.com:443	quic :443, p=1.000000, expires 2016-04-29 12:57:45
encrypted-tbn2.gstatic.com:443	quic :443, p=1.000000, expires 2016-04-29 12:57:45
encrypted-tbn0.gstatic.com:443	quic :443, p=1.000000, expires 2016-04-29 12:57:45
docs.google.com:443	quic :443, p=1.000000, expires 2016-04-28 14:07:21
0.docs.google.com:443	quic :443, p=1.000000, expires 2016-04-28 14:07:03
0.talkgadget.google.com:443	quic :443, p=1.000000, expires 2016-04-28 14:07:21
calendar.google.com:443	quic :443, p=1.000000, expires 2016-04-28 12:24:30
0.client-channel.google.com:443	quic :443, p=1.000000, expires 2016-04-28 12:22:24
2.client-channel.google.com:443	quic :443, p=1.000000, expires 2016-04-28 12:22:40
gm1.ggpht.com:443	quic :443, p=1.000000, expires 2016-04-28 12:22:10
maps.google.com:443	quic :443, p=1.000000, expires 2016-04-22 14:54:51
1.client-channel.google.com:443	quic :443, p=1.000000, expires 2016-04-21 15:48:51
support.google.com:443	quic :443, p=1.000000, expires 2016-04-21 15:01:53
storage.googleapis.com:443	quic :443, p=1.000000, expires 2016-04-21 13:29:33
img.youtube.com:443	quic :443, p=1.000000, expires 2016-04-21 10:22:56
myaccount.google.com:443	quic :443, p=1.000000, expires 2016-04-21 09:05:53
csi.gstatic.com:443	quic :443, p=1.000000, expires 2016-04-21 09:05:53
security.google.com:443	quic :443, p=1.000000, expires 2016-04-21 09:05:09
translate.googleapis.com:443	quic :443, p=1.000000, expires 2016-04-20 11:00:58
developers.google.com:443	quic :443, p=1.000000, expires 2016-04-19 14:32:14
groups.google.com:443	quic :443, p=1.000000, expires 2016-04-19 14:30:02
stats.g.doubleclick.net:443	quic :443, p=1.000000, expires 2016-04-18 19:13:25
googleads.g.doubleclick.net:443	quic :443, p=1.000000, expires 2016-04-18 19:10:35
chrome.google.com:443	quic :443, p=1.000000, expires 2016-04-18 18:55:11
www.googleadservices.com:443	quic :443, p=1.000000, expires 2016-04-18 18:54:42
2542116.fls.doubleclick.net:443	quic :443, p=1.000000, expires 2016-04-18 18:54:40
drive.google.com:443	quic :443, p=1.000000, expires 2016-04-28 12:24:04

ANALYZING QUIC

- Wireshark:

QUIC dissector =>



ANALYZING & DEFENDING - INSPECT

- H2
  - Doable if they aren’t changing the implementation
    - Look for non-typical behavior
      - Non-monotonous or non-increasing stream IDs
      - Strange content sent over control streams

- QUIC
  - Difficult due to crypto setup, likely requires new tools

CONCLUSIONS

(WHAT DOES IT MEAN?)

CONCLUSIONS – FUTURE WORK

- Other protocols
- Web RTC
- Extended application layer multiplexing
- Multipath QUIC, QUIC FEC

CONCLUSIONS - SUMMARY

- Tools MUST keep up with tech
- If tools can’t, then people must be aware
- Even if tools and people are away, playtime is over.

BRIEF TAKEAWAYS - SOUNDBYTES

- Technology is moving faster and faster:
  - Increasingly driven by large vendors, not standards bodies
  - Network security technology is surprisingly unaware of many application layer techniques
- Get ready for userspace network stacks
- Get ready for a lot more context heavy, encrypted, and multiplexed communications

Soundbytes

- HTTP/2 and QUIC provide enhanced user experience, making sites load faster and smoother than ever before
- HTTP2 is already bigger than IPv6, QUIC is already Bigger than MPTCP
  - > 1 billion devices using these technologies
- These protocols complicate network security
  - Designed to be more private than the legacy Internet
  - Security tools do not understand them
  - Even if security tools understand them, they offer so much more complexity that an attacker can hide in

QUESTIONS

Catherine (Kate) Pearce
Twitter: @secvalve
katpearc@cisco.com

Carl Vincent
carvince@cisco.com

REFERENCES 1 – MPTCP

- [MPTCP-A] – Pearce, C and Thomas, P - Multipath TCP: Pwning Today's Networks with Tomorrow's Protocols - https://www.blackhat.com/docs/us-14/materials/us-14-Pearce-Multipath-TCP-Breaking-Todays-Networks-With-Tomorrows-Protocols.pdf (slides) and https://www.blackhat.com/docs/us-14/materials/us-14-Pearce-Multipath-TCP-Breaking-Todays-Networks-With-Tomorrows-Protocols-WP.pdf (Paper)
- [MPTCP-B] – Pearce, C - *Multipath Madness, MPTCP, and Beyond - feat HTTP evasive fragmentation* – Hushcon East 2015, Kiwicon 9 - https://kiwicon.org/the-con/talks/#e206
- [MPTCP-C] – Pearce, C & Zeadally, S - *Ancillary Impacts of Multipath TCP on Current and Future Network Security* - http://www.computer.org/csdl/mags/ic/2015/05/mic2015050058-abs.html
- [MPTCP-D] Barré, S., Paasch, C. & Bonaventure, O., 2011. Multipath TCP: from theory to practice. Netw. 2011, pp.1–42. Available at: http://link.springer.com/chapter/10.1007/978-3-642-20757-0_35 http://datatracker.ietf.org/doc/draft-barre-mptcp-impl/ .
- [MPTCP-E] Bonaventure, O., 2012. An overview of Multipath TCP. ; login Mag. …, pp.17–23. Available at: http://dial.academielouvain.be/handle/boreal:114081 .
- [MPTCP-F] Raiciu, C. et al., 2012. How hard can it be? designing and implementing a deployable multipath TCP. NSDI, (1). Available at: https://www.usenix.org/system/files/conference/nsdi12/nsdi12-final125.pdf .

REFERENCES 2 – HTTP/2

- [HTTP2-A] Akamai HTTP/2 Demo - https://http2.akamai.com/demo
- [HTTP2-B] HTTP/2 - http://http2.github.io/http2-spec/
- [HTTP2-C] caniuse HTTP/2 - http://caniuse.com/#feat=http2
- [HTTP2-D] chrome platform status - https://www.chromestatus.com/features/5152586365665280
- [HTTP2-E] IE platform status - https://dev.windows.com/en-us/microsoft-edge/platform/status/http2
- [HTTP2-F] RFC 7540 - Hypertext Transfer Protocol Version 2 (HTTP/2) - https://tools.ietf.org/html/rfc7540
- [HTTP2-G] http/2 spec - http://http2.github.io/http2-spec/index.html
- [HTTP2-H] HTTP2 – Daniel Steinberg - https://daniel.haxx.se/http2/http2-v1.2.pdf
- [HTTP2-I] HTTP2 Explained – Daniel Steinberg - http://www.sigcomm.org/sites/default/files/ccr/papers/2014/July/0000000-0000017.pdf
- [HTTP2-J] Scapy-http2 - https://github.com/alexmgr/scapy-http2
- [HTTP2-k] HTTP/2 For Web Application Developers – nginx - https://www.nginx.com/wp-content/uploads/2015/09/NGINX_HTTP2_White_Paper_v4.pdf
- [HTTP2-L] Attacking HTTP2 Implementations - https://yahoo-security.tumblr.com/post/134549767190/attacking-http2-implementations and http://www.slideshare.net/JohnVillamil/attacking-http2-implementations-1
- [HTTP2-M] Sans Infosec Handlers Diary Blog – RFC 7540 – HTTP/2 Protocol - https://isc.sans.edu/diary/RFC+7540+-+http2+protocol/19799
- [HTTP2-N] http2fuzz – HTTP2 Fuzzer Written in golang - https://github.com/c0nrad/http2fuzz
- [HTTP2-O] Tools for debugging, testing and using HTTP/2 - https://blog.cloudflare.com/tools-for-debugging-testing-and-using-http-2/
- [HTTP2-P] HTTP/2 Considerations and Tradeoffs - https://insouciant.org/tech/http-slash-2-considerations-and-tradeoffs/
- [HTTP2-Q] Shodan – Tracking HTTP/2 Adoption - https://blog.shodan.io/tracking-http2-0-adoption/

REFERENCES 3 - QUIC

- [QUIC-A] Hamilton, R; Iyengar, J; Swett, I; Wilk, A - “QUIC: A UDP-Based Secure and Reliable Transport for HTTP/2 - draft-tsvwg-quic-protocol-02” - https://tools.ietf.org/html/draft-tsvwg-quic-protocol-02
- [QUIC-B] Chromium Source Tree for QUIC - https://chromium.googlesource.com/chromium/src/+/master/net/quic
- [QUIC-C] Playing with QUIC - https://www.chromium.org/quic/playing-with-quic
- [QUIC-D] QUIC: next generation multiplexed transport over UDP - https://www.youtube.com/watch?v=hQZ-0mXFmk8
- [QUIC-E] QUIC FAQ For Geeks - https://docs.google.com/document/d/1lmL9EF6qKrk7gbazY8bIdvq3Pno2Xj_l_YShP40GLQE/edit?pref=2&pli=1
- [QUIC-F] QUIC Design Document - https://docs.google.com/document/d/1RNHkx_VvKWyWg6Lr8SZ-saqsQx7rFV-ev2jRFUoVD34/edit?pref=2&pli=1
- [QUIC-G] Pearce, C – *Multipathed, Multiplexed, Multilateral Transport Protocols - Decoupling transport protocols from what's below* – 2016 Asia Pacific Regional Internet Conference on Operational Technologies (APRICOT 2016) https://conference.apnic.net/data/41/cpearce-multipath-_-apricot-3_1456107769.pdf
- [QUIC-H] Pearce, C – *Transport Futures: Moving Targets and Multi-dimensional Fragmentation - Multipathed, Multiplexed and Multilateral Network Security* - Australian Cyber Security Conference 2016 - http://acsc2016.com.au/program/?IntCatId=27&IntContId=7741#futures
- [QUIC-I] Block QUIC Protocol, Squid Knowledgebase - http://wiki.squid-cache.org/KnowledgeBase/Block%20QUIC%20protocol
- [QUIC-J] libquic - https://github.com/devsisters/libquic
- [QUIC-K] goquic - https://github.com/devsisters/goquic
- [QUIC-L] Fortinet Technical Note: How To Block/Disable QUIC - http://kb.fortinet.com/kb/documentLink.do?externalID=FD36680
- [QUIC-M] Nancy Sepuran, Faster Protocols and the Future of Next-Generation Firewalls - http://www.endtoend.com/faster-protocols-future-next-generation-firewalls/
- [QUIC-N] - https://www.chromestatus.com/features#quic
- [QUIC-O] On the Security of TLS 1.3 and QUIC Against Weaknesses in PKCS#1 v1.5 Encryption - Jaeger, T et al. - https://www.nds.rub.de/media/nds/veroeffentlichungen/2015/08/21/Tls13QuicAttacks.pdf
- [QUIC-P] (Slides) On the Security of TLS 1.3 (and QUIC) Against Weaknesses in PKCS#1 v1.5 Encryption - https://www.internetsociety.org/sites/default/files/T8-jager.pdf
- [QUIC-Q] QUIC and TLS – Adam Langley - https://www.ietf.org/proceedings/92/slides/slides-92-saag-5.pdf
- [QUIC-R] QUIC Crypto - https://docs.google.com/document/d/1g5nIXAIkN_Y-7XJW5K45IblHd_L2f5LTaDUDwvZ5L6g/edit

REFERENCES 4 - OTHER

- [Other-1] TLS 1.3 SPEC - https://tlswg.github.io/tls13-spec/
- [Other-2] Tim Taubert - MORE PRIVACY, LESS LATENCY
- Improved Handshakes in TLS version 1.3 - https://timtaubert.de/blog/2015/11/more-privacy-less-latency-improved-handshakes-in-tls-13/