nDPI performance and QUIC

Ivan Nardi

ntopConf '22

June, 23-24 · Milan

Agenda

- nDPI performance:
  - testing nDPI with existing probes with REAL traffic

- QUIC: let's demystify this new protocol

Who am I?

- Ivan Nardi, @ AI2M:
  - lawful interception, investigation analysis, big data retention
  - voice/IP metadata collection, processing and reporting
  - network probes and DPI

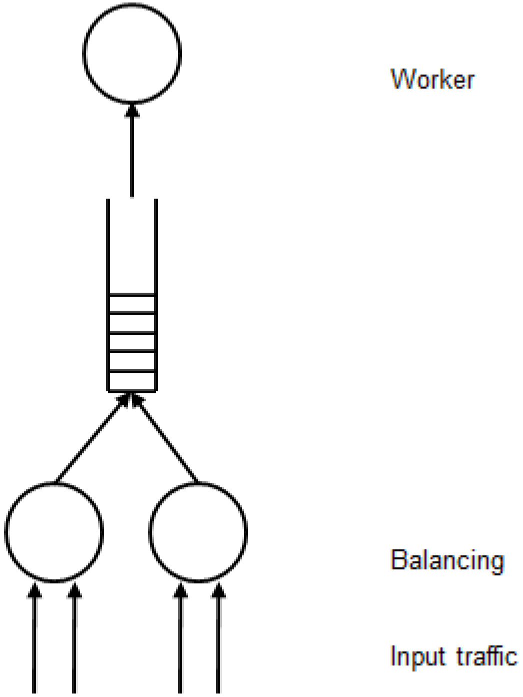
- ivan@ai2m.eu

nDPI: integration on existing probes

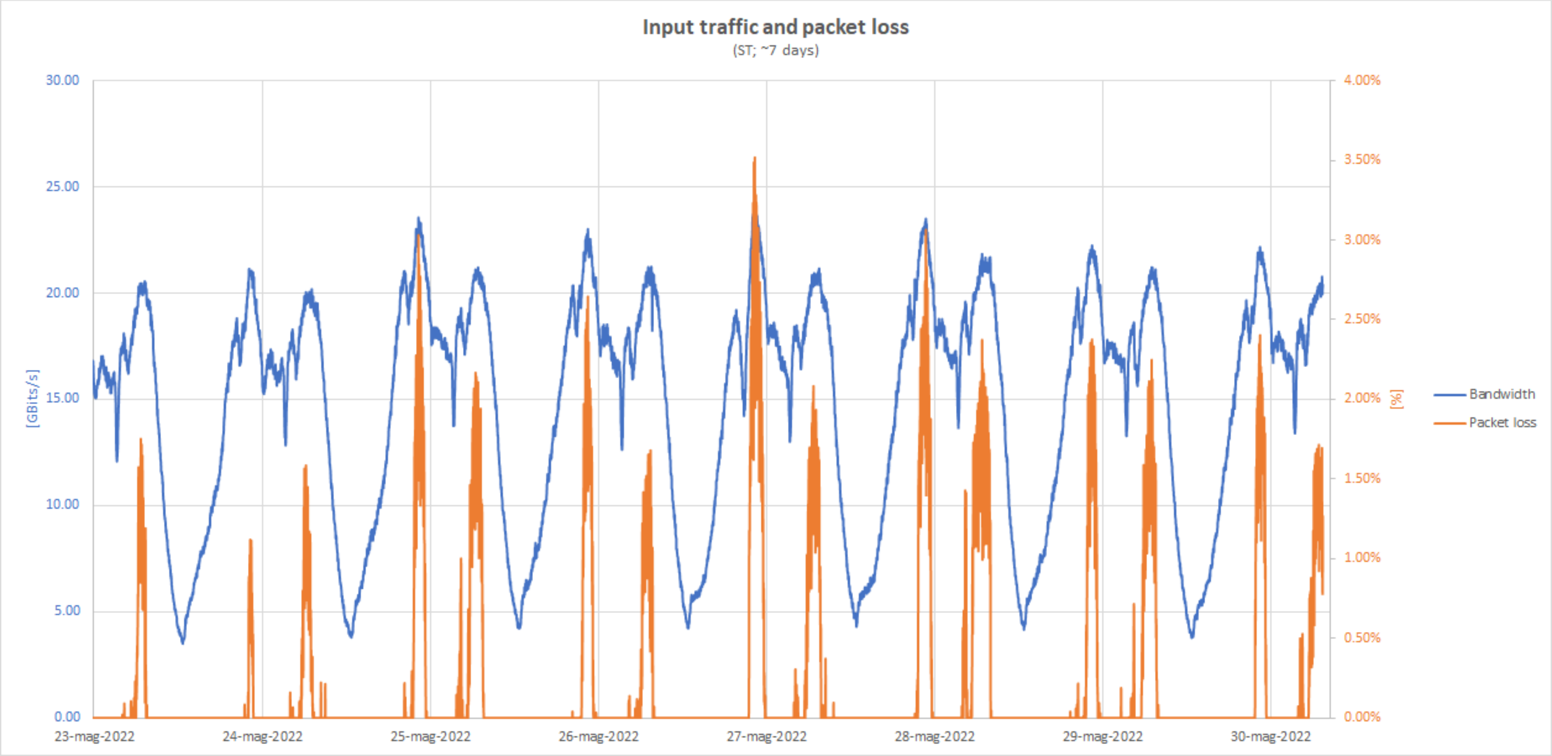
- Software:
  - nDPI (dev branch, 2560260a) with default configuration
  - all ~300 protocols enabled + ~20 other protocols
- Full metadata extraction. Exceptions:
  - no DNS sub-classification
  - no parsing of HTTP replies
  - no JA3/JA3S calculation
- Some private patches: integration, performance, statistics, ipv6

nDPI: single thread performance

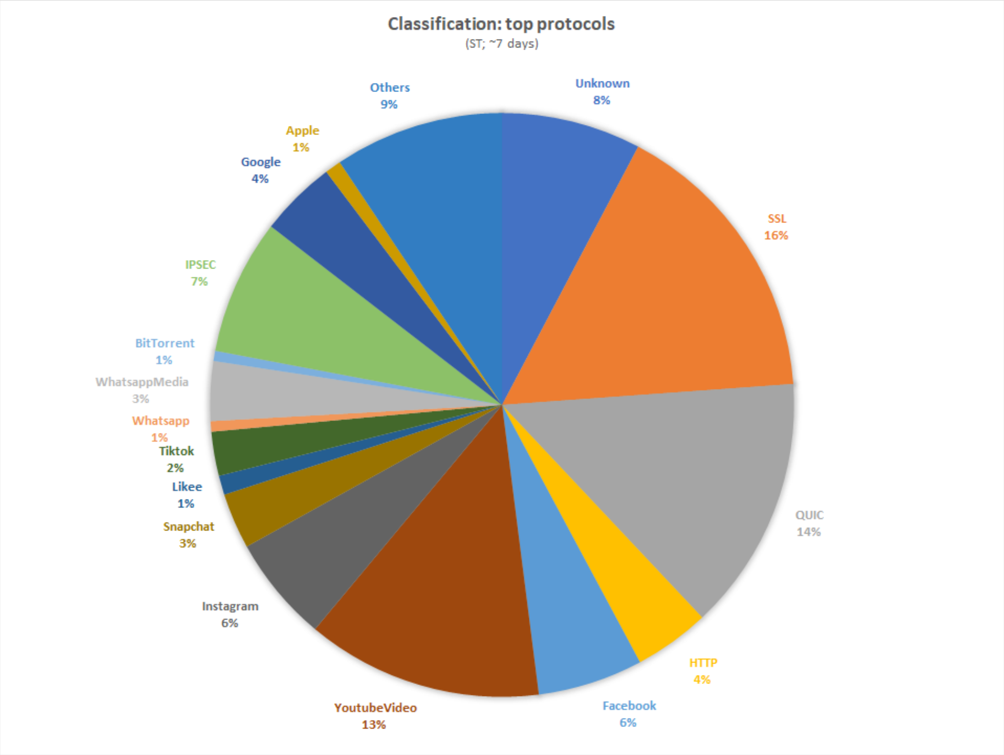
- Environment (single-thread)
  - Intel Xeon E5-2690 @ 2.90GHz (2012!)
  - Intel X710 4x10Gb
  - 4 * 10Gb links
- Traffic: residential (fiber & ADSL), mobile, enterprise



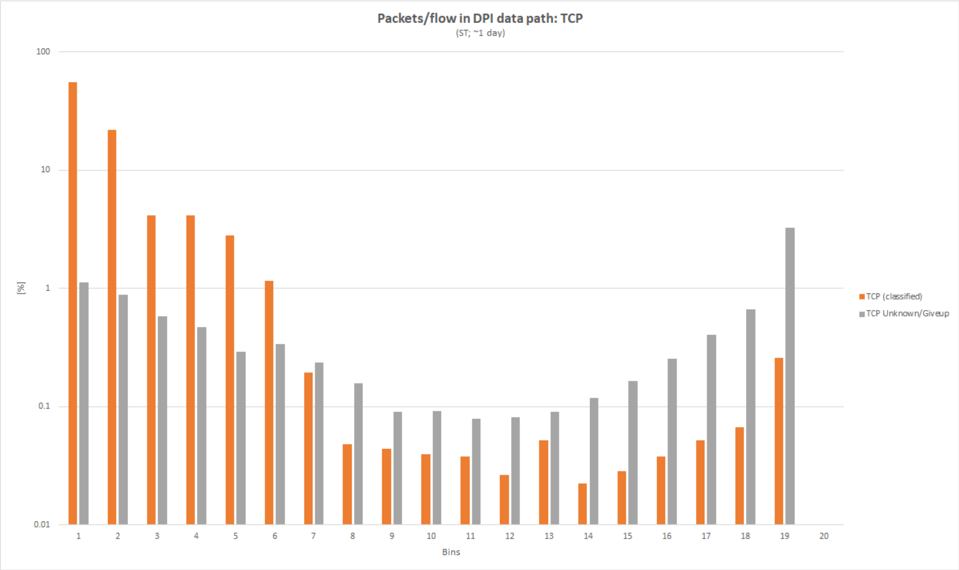
Input traffic and packet loss

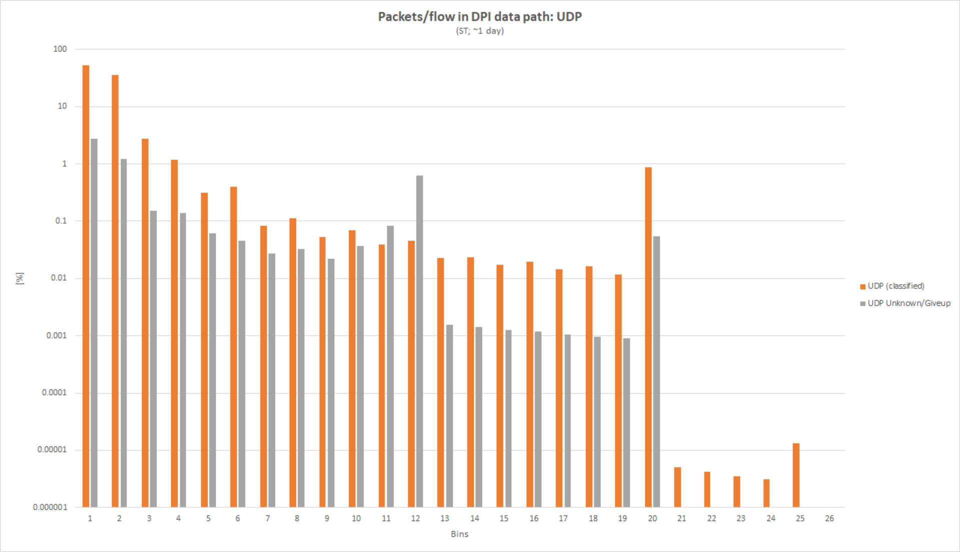


Classification: top protocols

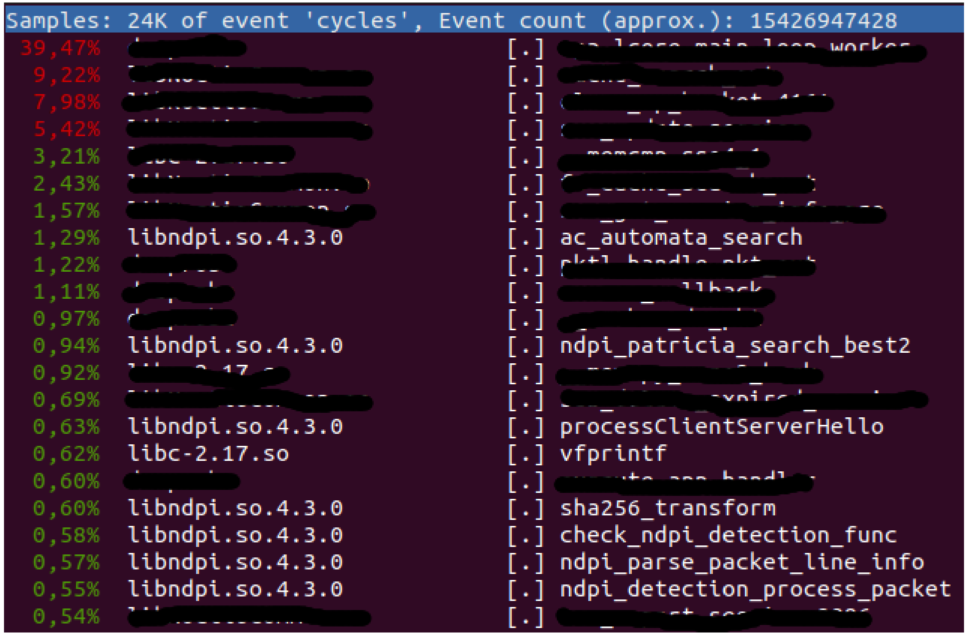


Packets/flow in DPI data path

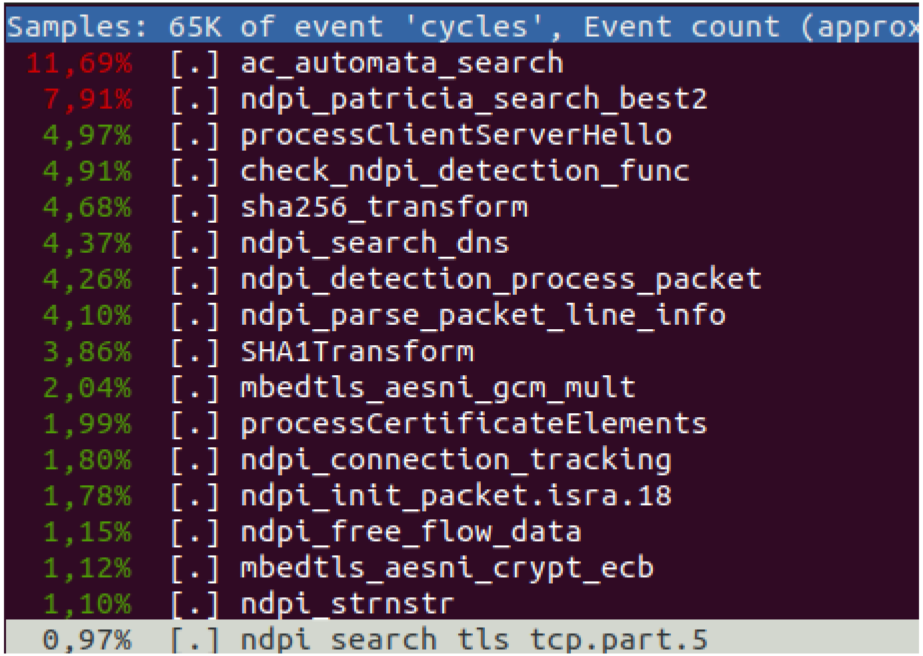




Profiling via perf



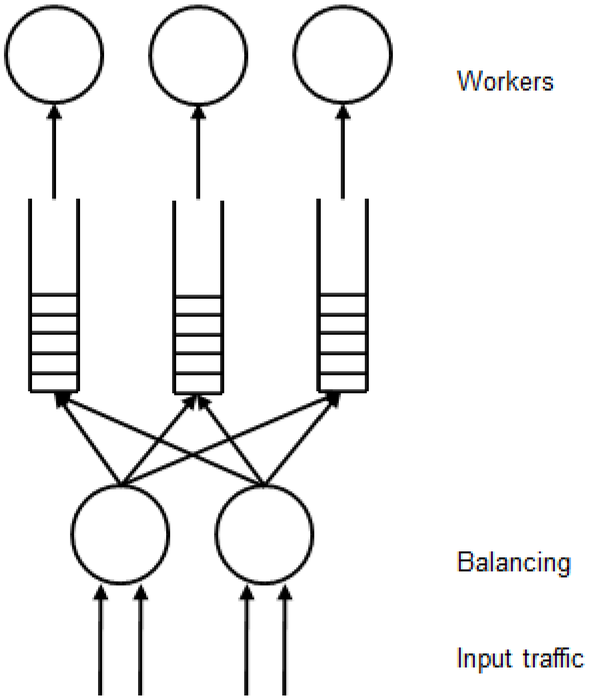
```
Samples: 24K of event 'cycles', Event count (approx.): 15426947428
39,47%  [redacted]            [.] [redacted]
 9,22%  [redacted]            [.] [redacted]
 7,98%  [redacted]            [.] [redacted]
 5,42%  [redacted]            [.] [redacted]
 3,21%  [redacted]            [.] [redacted]
 2,43%  [redacted]            [.] [redacted]
 1,57%  [redacted]            [.] [redacted]
 1,29%  libndpi.so.4.3.0      [.] ac_automata_search
 1,22%  [redacted]            [.] [redacted]
 1,11%  [redacted]            [.] [redacted]
 0,97%  [redacted]            [.] [redacted]
 0,94%  libndpi.so.4.3.0      [.] ndpi_patricia_search_best2
 0,92%  [redacted]            [.] [redacted]
 0,69%  [redacted]            [.] [redacted]
 0,63%  libndpi.so.4.3.0      [.] processClientServerHello
 0,62%  libc-2.17.so          [.] vfprintf
 0,60%  [redacted]            [.] [redacted]
 0,60%  libndpi.so.4.3.0      [.] sha256_transform
 0,58%  libndpi.so.4.3.0      [.] check_ndpi_detection_func
 0,57%  libndpi.so.4.3.0      [.] ndpi_parse_packet_line_info
 0,55%  libndpi.so.4.3.0      [.] ndpi_detection_process_packet
 0,54%  [redacted]            [.] [redacted]
```



```
Samples: 65K of event 'cycles', Event count (approx
 11,69%  [.] ac_automata_search
  7,91%  [.] ndpi_patricia_search_best2
  4,97%  [.] processClientServerHello
  4,91%  [.] check_ndpi_detection_func
  4,68%  [.] sha256_transform
  4,37%  [.] ndpi_search_dns
  4,26%  [.] ndpi_detection_process_packet
  4,10%  [.] ndpi_parse_packet_line_info
  3,86%  [.] SHA1Transform
  2,04%  [.] mbedtls_aesni_gcm_mult
  1,99%  [.] processCertificateElements
  1,80%  [.] ndpi_connection_tracking
  1,78%  [.] ndpi_init_packet.isra.18
  1,15%  [.] ndpi_free_flow_data
  1,12%  [.] mbedtls_aesni_crypt_ecb
  1,10%  [.] ndpi_strnstr
  0,97%  [.] ndpi_search_tls_tcp.part.5
```

nDPI performance: multiple threads

- Environment (multi-threads)
  - 2 x Intel Xeon E5-2697A v4 @ 2.60GHz, 16 core (2016)
  - Intel X710 4x10Gb
  - 24 * 10Gb links
- Results:
  - no packet loss; same classifications as ST; no sharing data



nDPI: performance

- Conclusions:
  - nDPI might be extremely cheap (from a resources POV)
  - nDPI has optimal scaling performance

QUIC

https://www.smashingmagazine.com/2021/08/http3-core-concepts-part1

https://www.youtube.com/watch?v=jQ1GCkhwGTg

QUIC: what?

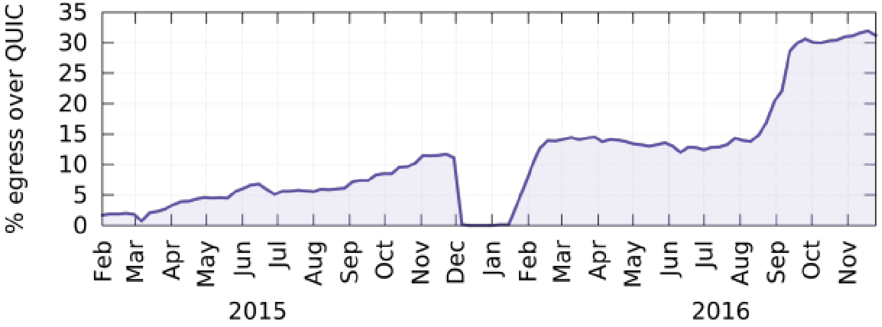
- First things first: thanks to @programmingart for allowing to use all these nice images

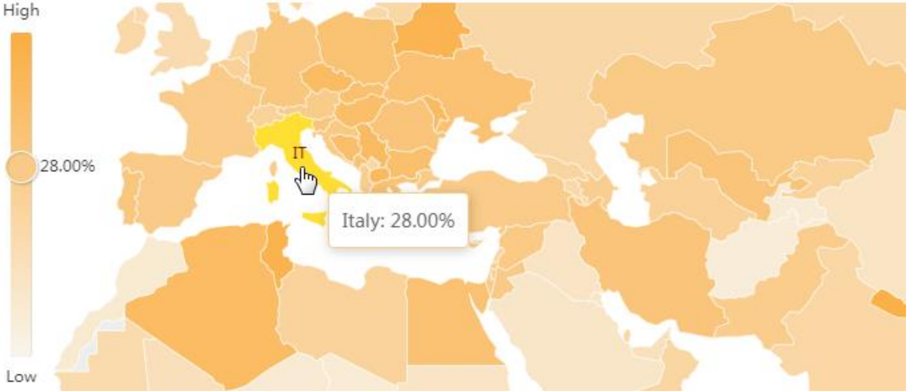
- "QUIC is a secure general-purpose transport protocol [and it] is secured using TLS" [RFC8999-9002][05/2021]

- Oversimplifying: QUIC = TCP + TLS over UDP

QUIC: who and since when?

- HTTP/3 over QUIC [RFC9114][06/2022]: HTTP traffic from browsers and mobile apps
  - All major browsers
  - All major CDNs: Fastly, Cloudflare, Akamai...
  - Biggest internet company: Google, FB, Snapchat





QUIC: who and since when?

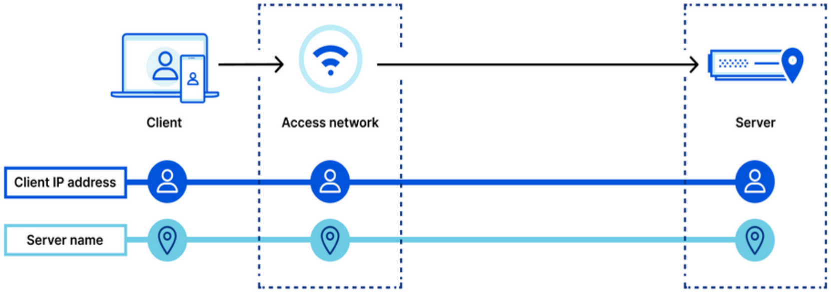
- DNS over QUIC [RFC9250, 05/2022]
  - DoH-DoT privacy + UDP latency
  - AdGuard deployed it on 12/2020[1]

- SMB over QUIC
  - Present in Windows 11 and Windows Server 2022[2]

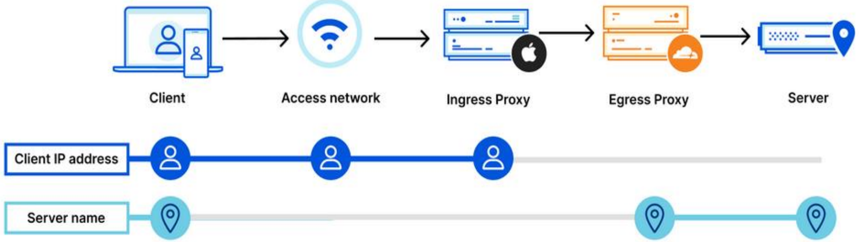
[1] https://adguard.com/en/blog/dns-over-quic.html
[2] https://docs.microsoft.com/en-us/windows-server/storage/file-server/smb-over-quic

QUIC: who and since when?

- ICloud Private Relay [12/2021]
  - Dual-hop architecture: no single party has access to both the user's IP address and SNI[1][2]



  - QUIC Proxy (MASQUE WG)[3]



[1] https://www.apple.com/privacy/docs/iCloud_Private_Relay_Overview_Dec2021.PDF
[2] https://blog.cloudflare.com/icloud-private-relay/
[3] https://datatracker.ietf.org/doc/html/draft-ietf-masque-connect-udp-12

QUIC: who and since when?

- RTP/RTCP/WEBRTC over QUIC
  - MoQ (Working group?)[1]
  - RUSH: Facebook Live Video Ingest [07/2021][2]
  - QUIC demultiplexing (like STUN/RTP/RTCP over UDP)[3]
  - Snapchat (video)calls [07/2020, at least]

[1] https://datatracker.ietf.org/meeting/113/materials/agenda-113-moq-06

[2] https://www.ietf.org/archive/id/draft-kpugin-rush-00.html

[3] https://www.ietf.org/archive/id/draft-ietf-avtcore-rfc7983bis-04.txt

QUIC: who and since when?

- Fortigate Url filter [05/2022]
    - Inspecting and blocking HTTP3 traffic depending on keyword match[1][2]

- BGP over QUIC[3]
- SSH over QUIC[4]

[1] https://docs.fortinet.com/document/fortigate/7.2.0/new-features/440398/inspecting-http3-traffic

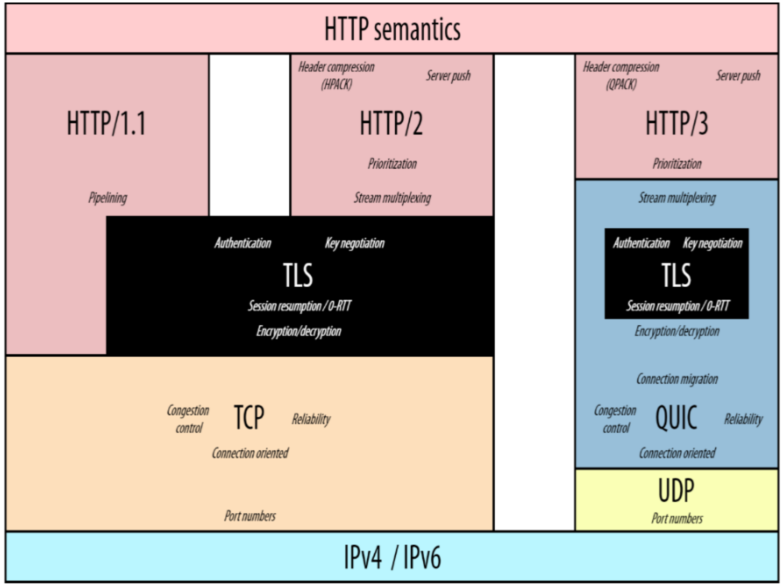
[2] https://www.youtube.com/watch?v=SI4OXspDuNI

[3] https://datatracker.ietf.org/doc/html/draft-chen-idr-bgp-over-quic-00.txt

[4] https://datatracker.ietf.org/doc/html/draft-bider-ssh-quic-09

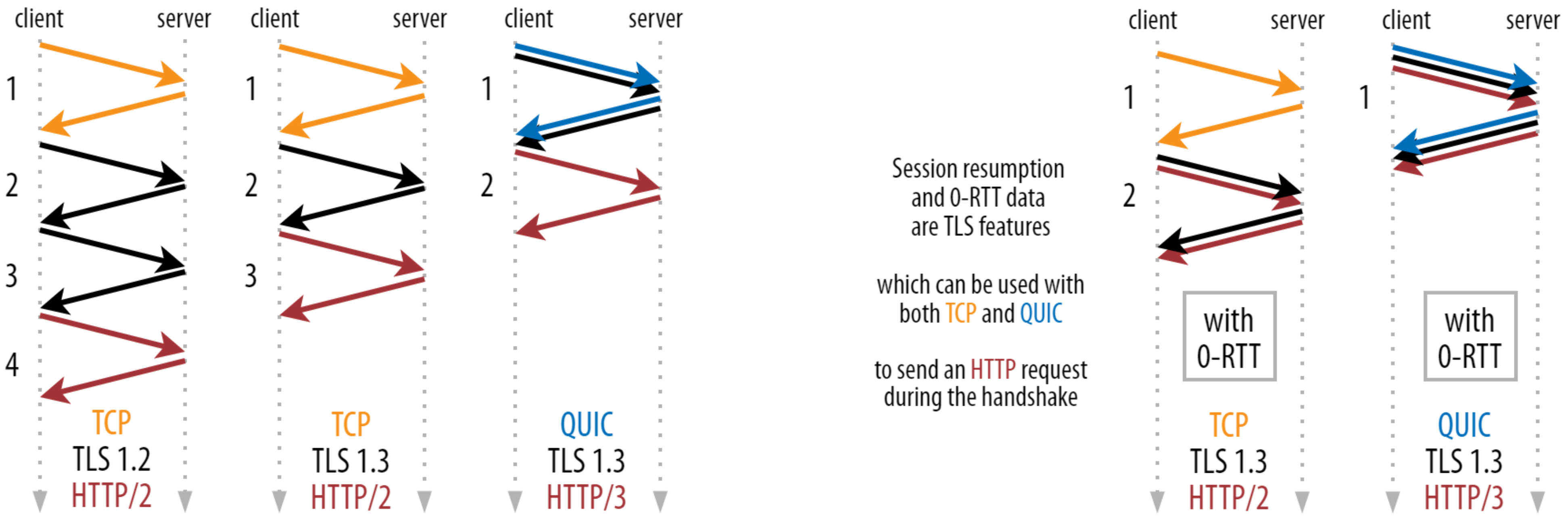
QUIC: what?

- Oversimplifying: QUIC = TCP + TLS over UDP
  - all TCP features: reliability, acknowledgements/retransmissions, a highly complex handshake, flow-control and congestion-control
  - all TLS features: encryption always on; no such thing like "plaintext QUIC"
  - it is built on top of UDP



QUIC: differences compared to TLS/TCP/UDP

- Connection set-up is faster



Session resumption
and 0-RTT data
are TLS features

which can be used with
both TCP and QUIC

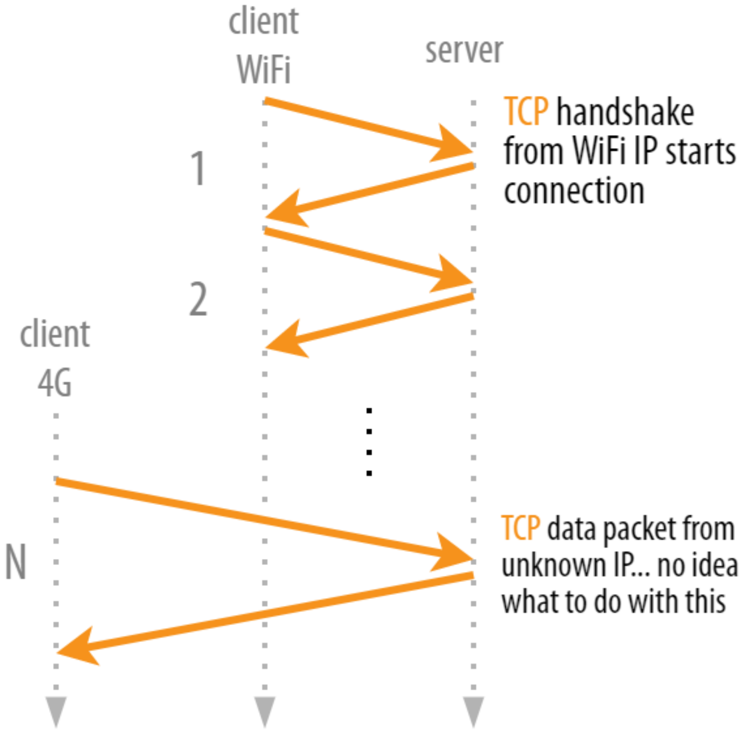
to send an HTTP request
during the handshake

QUIC: differences compared to TLS/TCP/UDP

- Better performance when data packets are lost
  - Supports for multiple independent byte streams (like SCTP)

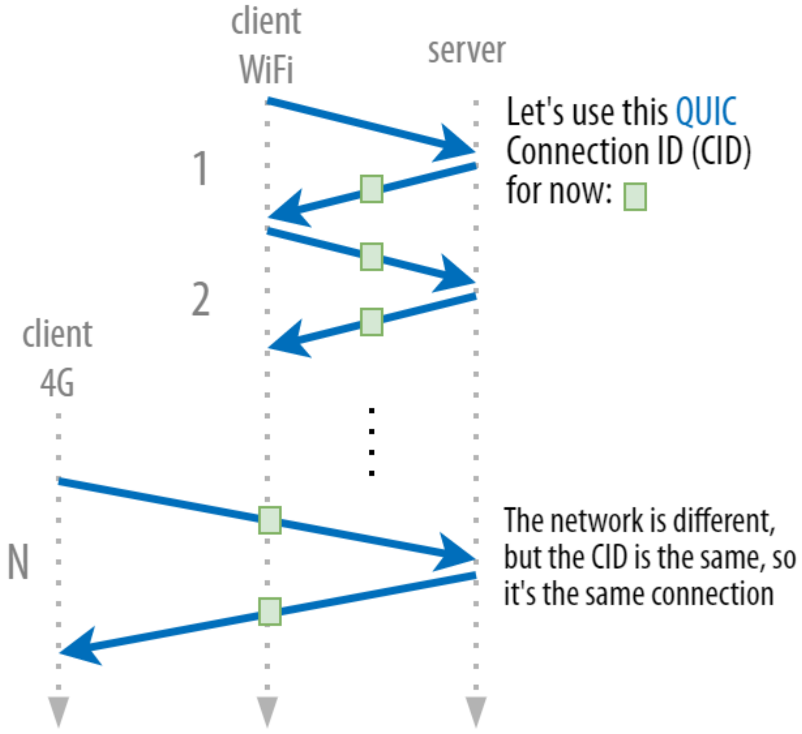
- Stable connections when networks change
  - Connection IDs (like GTP TEID or SCTP Verification Tag)
    - In TCP, connections are identified by the 5-tuple. So, if just one of those five parameters changes, the connection becomes invalid and needs to be re-established
    - In QUIC, a number is assigned to each connection and it uniquely identifies the connection between two endpoints.

QUIC: differences compared to TLS/TCP/UDP



TCP handshake from WiFi IP starts connection

TCP data packet from unknown IP... no idea what to do with this

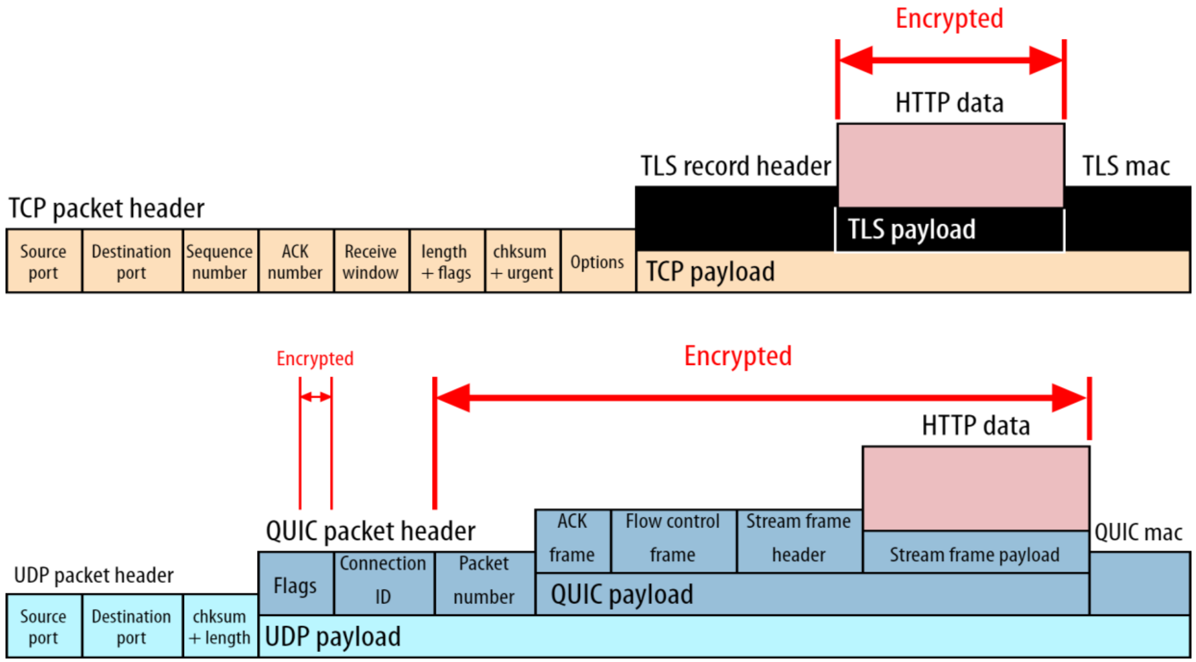


Let's use this QUIC Connection ID (CID) for now: 🟩

The network is different, but the CID is the same, so it's the same connection

QUIC: differences compared to TLS/TCP/UDP

- Deeply integration with TLS: user data and L4 fields are always encrypted



QUIC: advanced features

- QUIC is easier to improve and develop
  - Rapid deployment of QUIC modifications updating only the endpoints
  - Goal: avoid protocol ossification
- Connection migration: connection ID allows connections to survive changes to endpoint addresses (IP and/or port)
  - Nat rebinding or switching networks
- Multi-path: using multiple path at the same time [2019][1]
- Integrated logging facilities[2]

[1]https://datatracker.ietf.org/meeting/interim-2020-quic-02/materials/slides-interim-2020-quic-02-sessa-mpquic-use-cases-00.pdf

[2] https://datatracker.ietf.org/doc/html/draft-ietf-quic-qlog-main-schema

QUIC: conclusion

- Take a look at what’s happening on your networks at UDP/443
- We will see a lot of changes in network protocols in the next months/years

Thanks for your time. Questions?