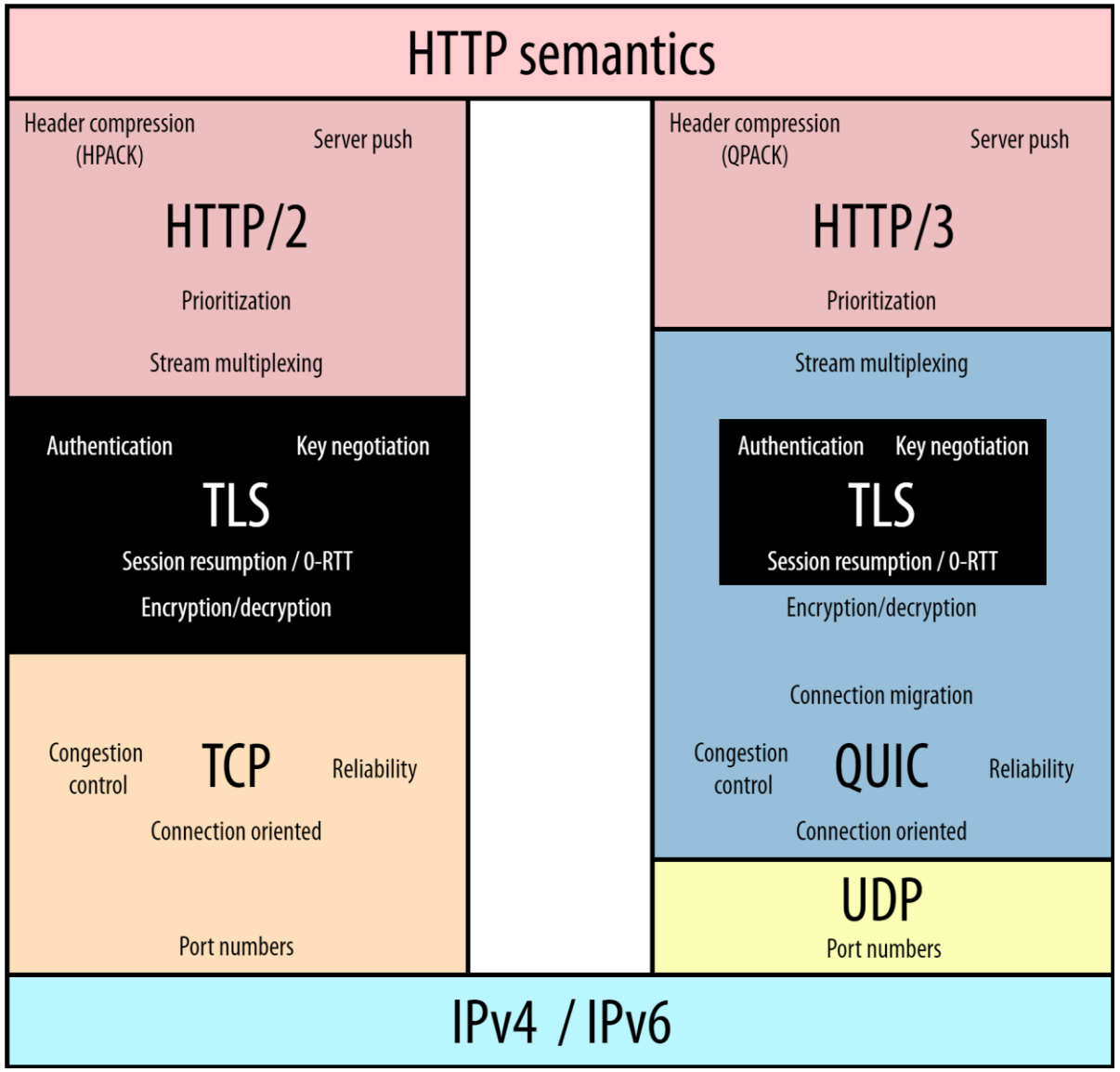
QUIC and HTTP/3 Security

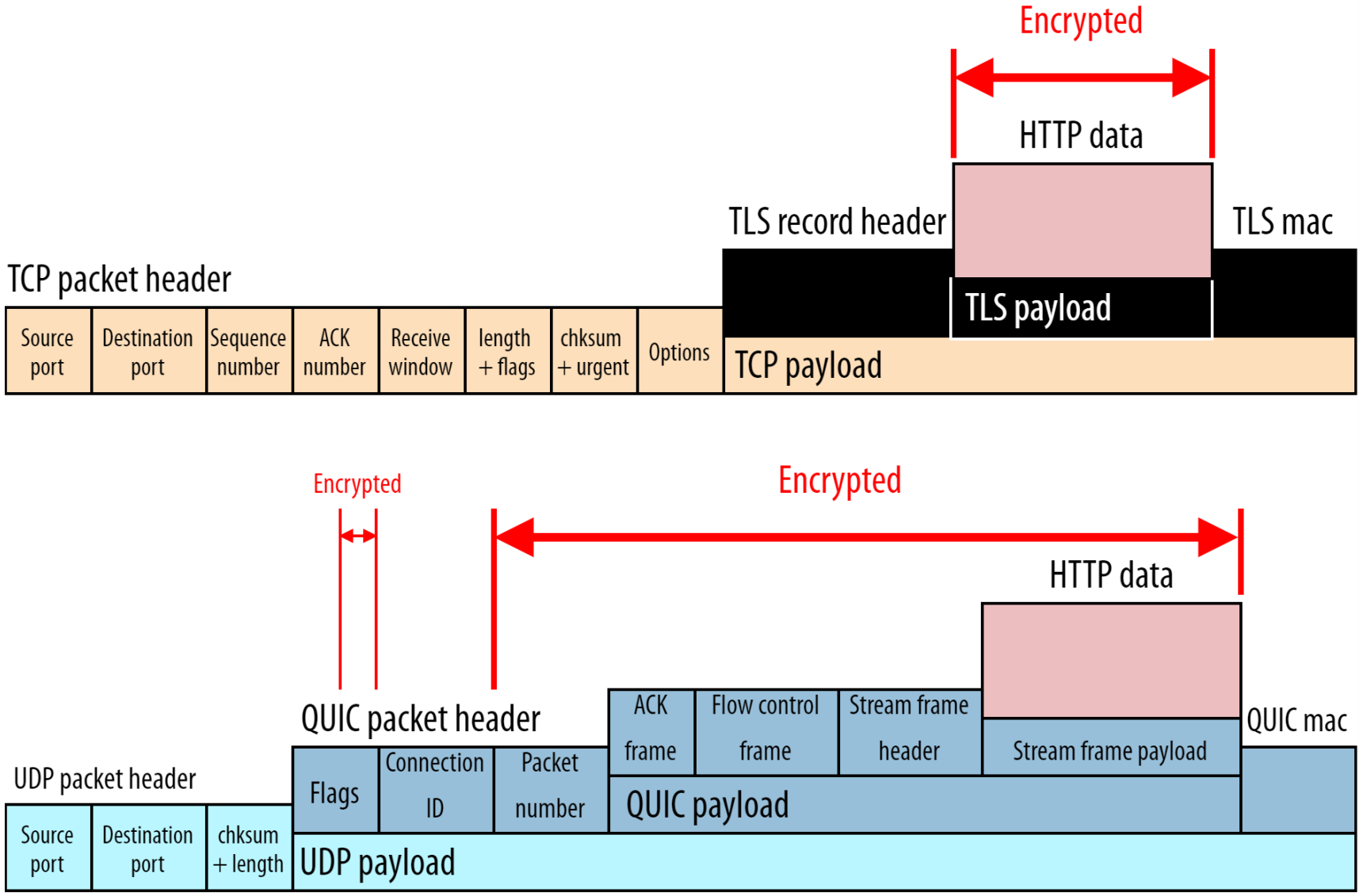
Robin Marx robin.marx@kuleuven.be

TCP+TLS vs QUIC



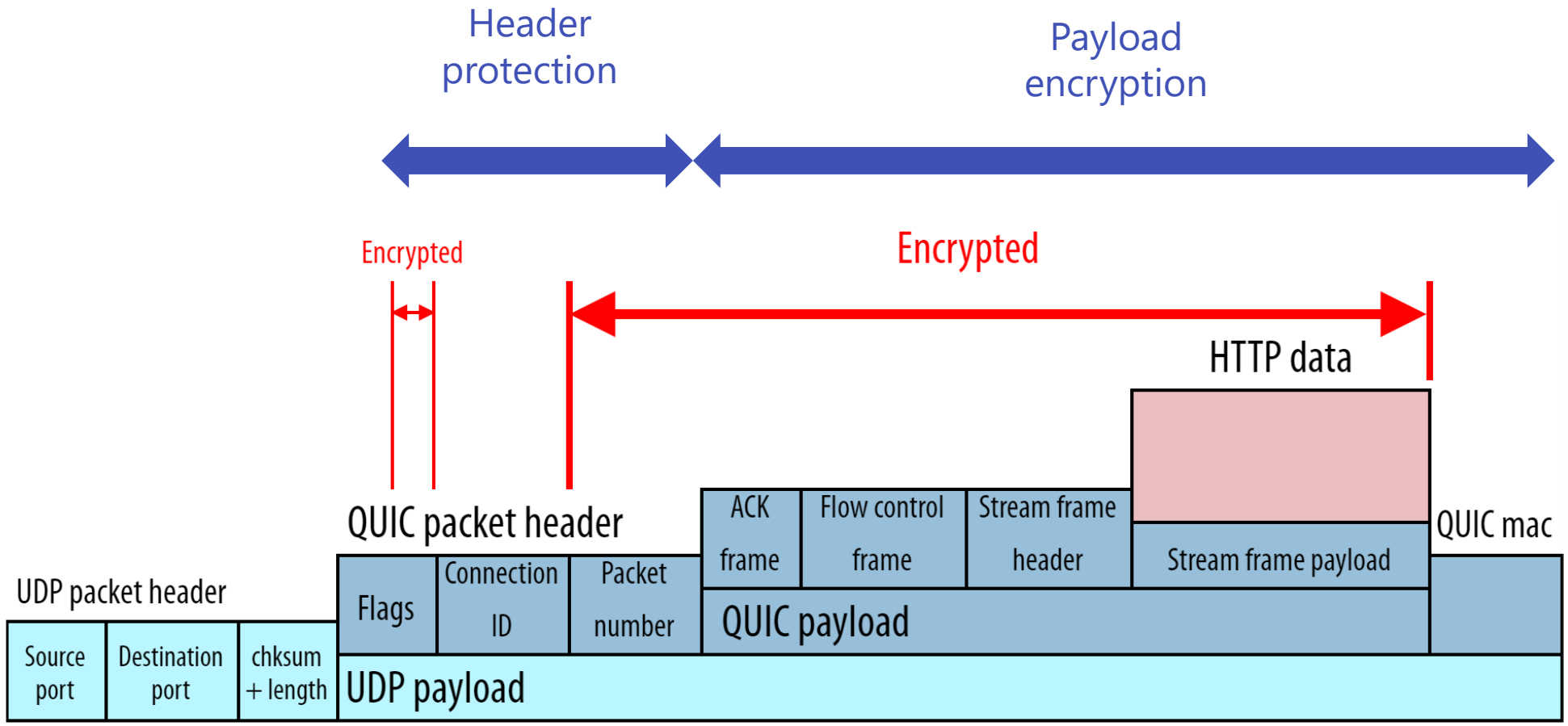
https://www.smashingmagazine.com/2021/08/http3-core-concepts-part1/

TCP+TLS vs QUIC



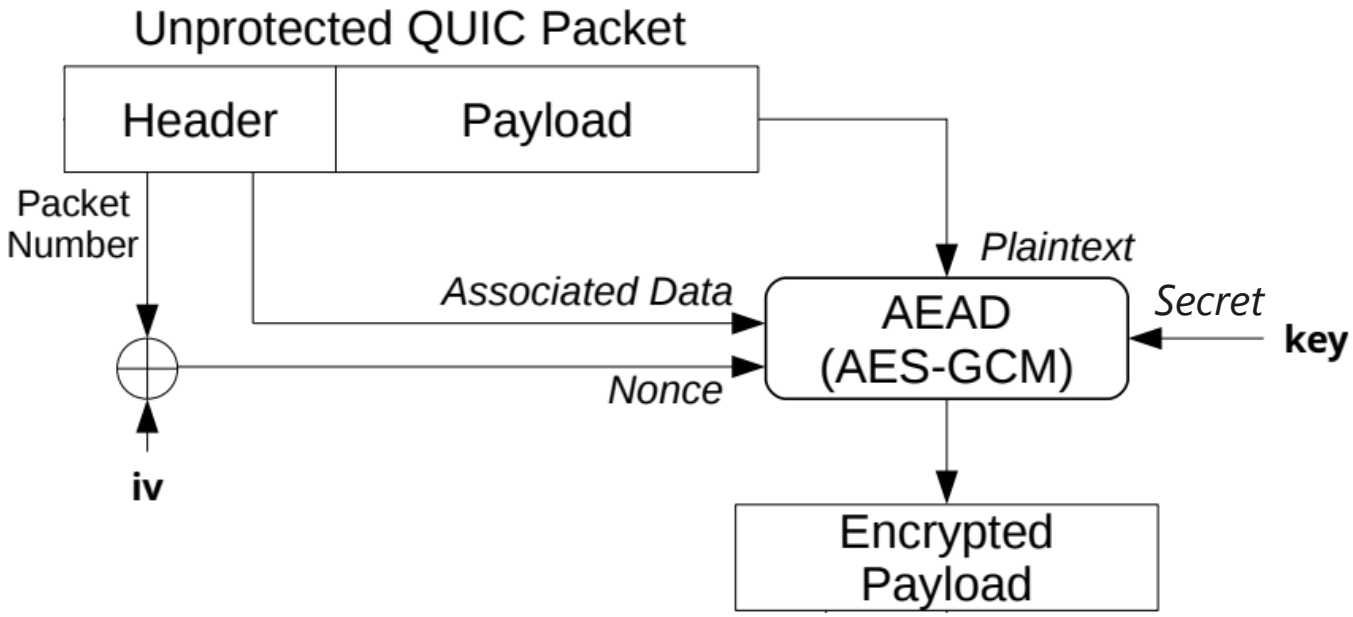
https://www.smashingmagazine.com/2021/08/http3-core-concepts-part1/

QUIC packets are “encrypted twice”



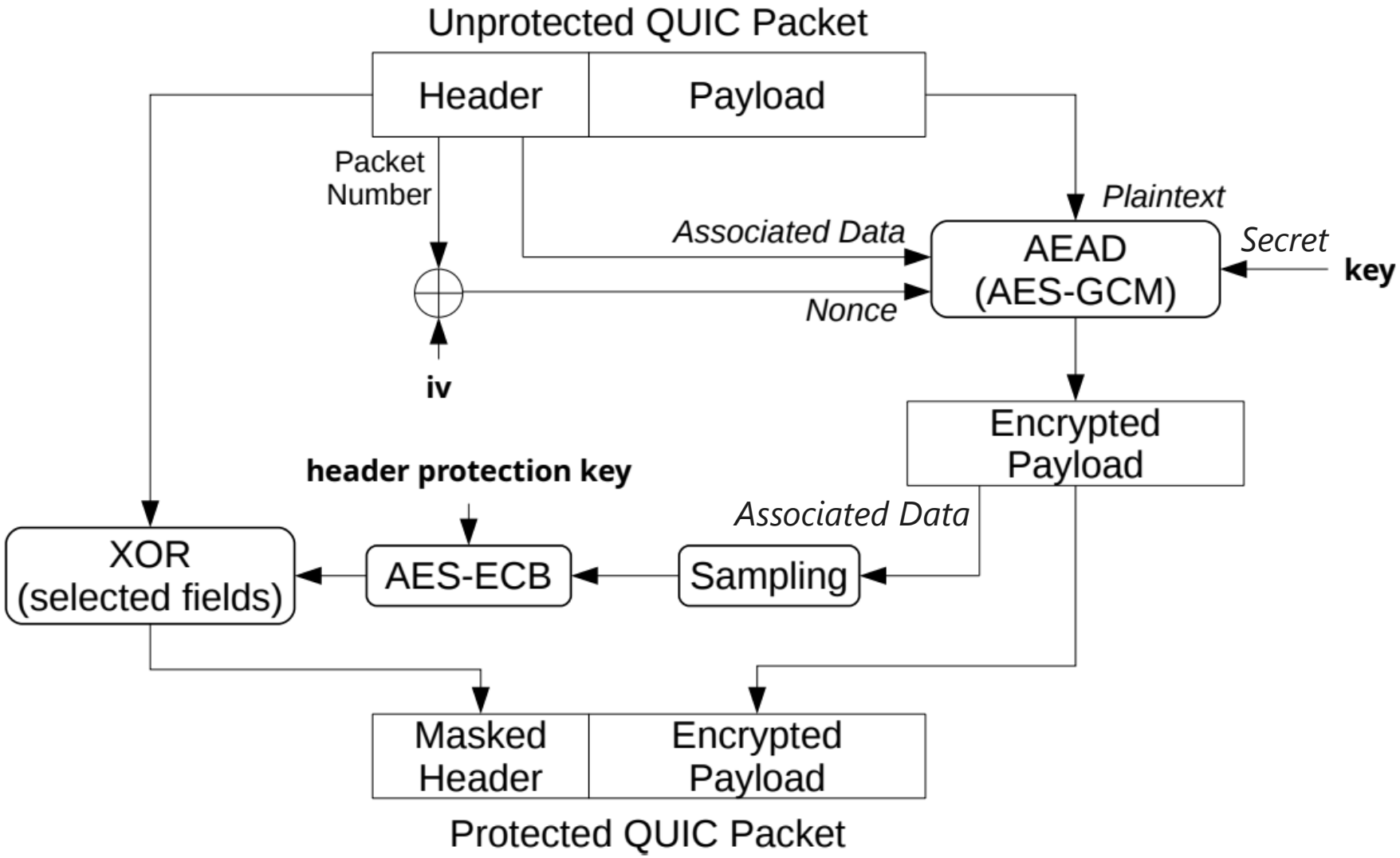
https://www.smashingmagazine.com/2021/08/http3-core-concepts-part1/

Payload encryption



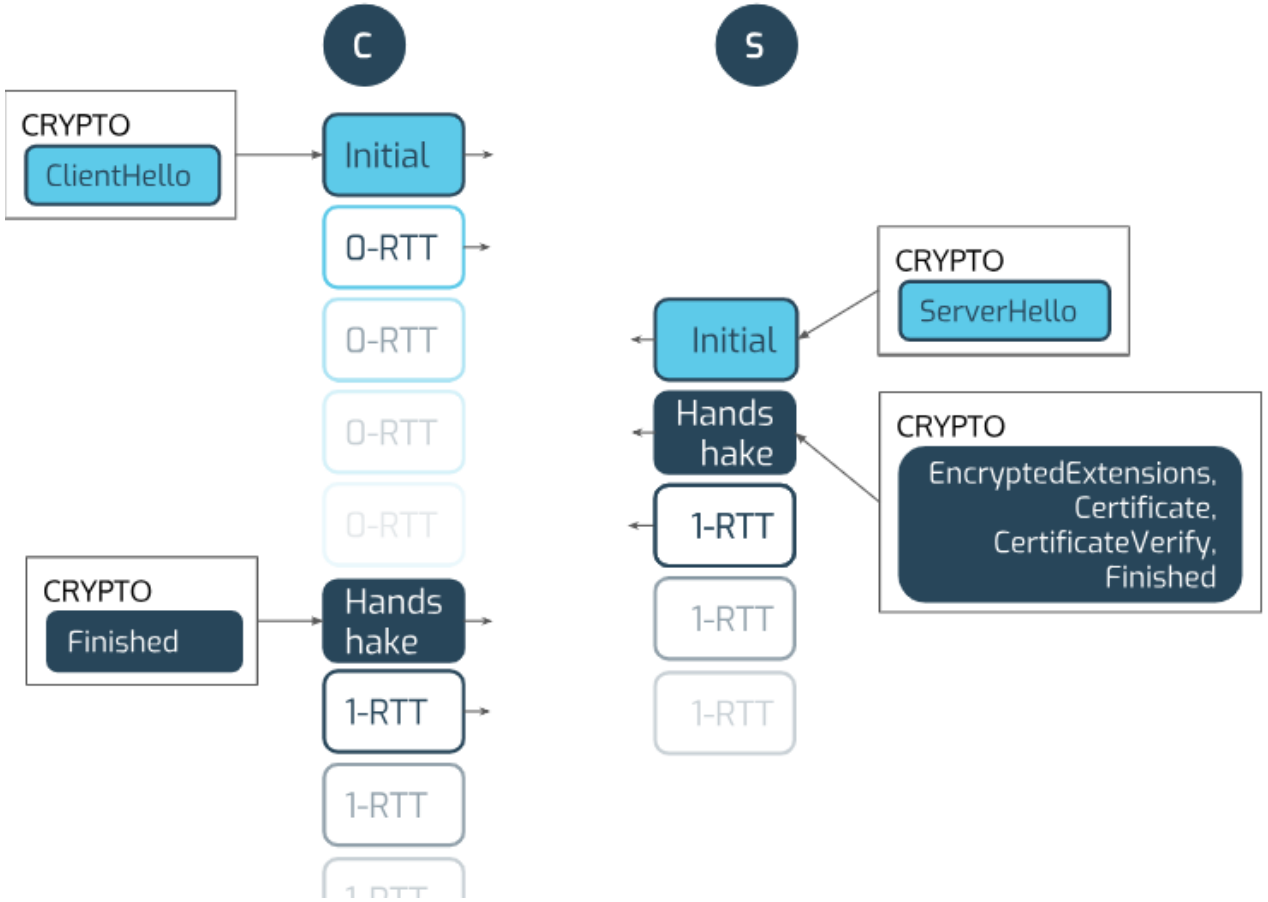
https://hal.archives-ouvertes.fr/hal-02468596/document

Header protection and Payload encryption



https://hal.archives-ouvertes.fr/hal-02468596/document

Combined Transport and Crypto handshake

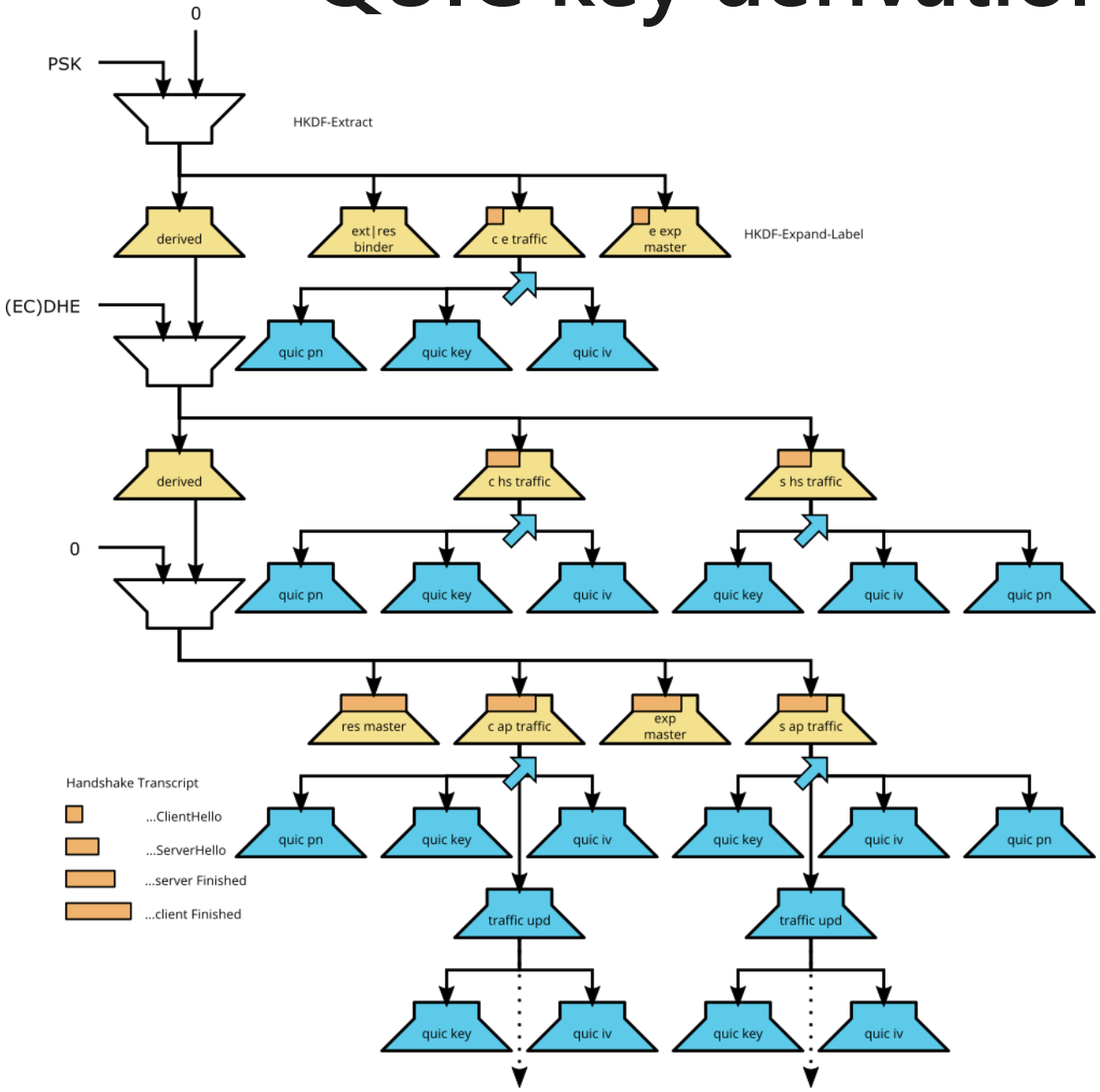


Main differences from TLS 1.3

- Hello's are "encrypted" (obfuscated)
- No EndOfEarlyData message
- No records!
  - Replaced by QUIC CRYPTO and STREAM frames
- Separate key derivations

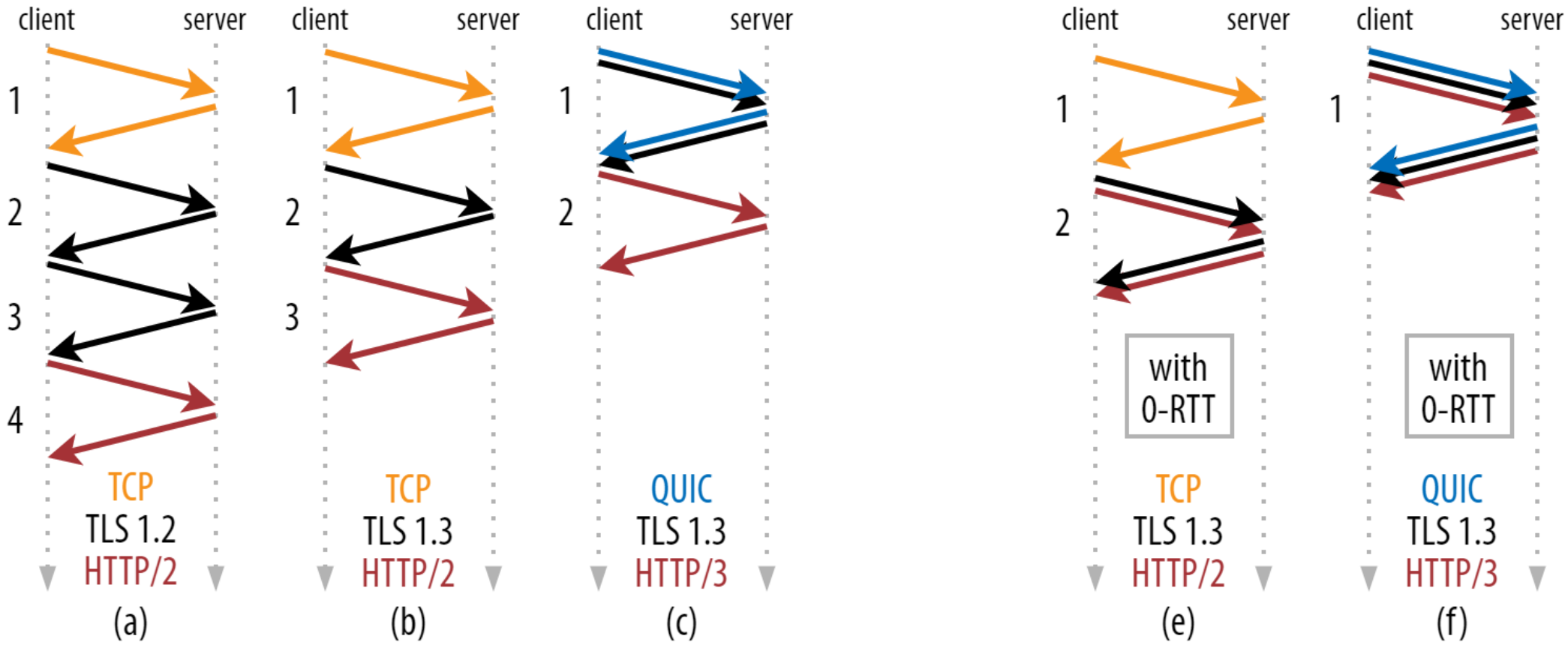
https://asnokaze.hatenablog.com/entry/2019/04/22/000927
https://docs.google.com/presentation/d/1OASDYIJlgSFg6hRkUjqdKfYTK1ZUk5VMGP3Iv2zQCI8

QUIC key derivation schedule



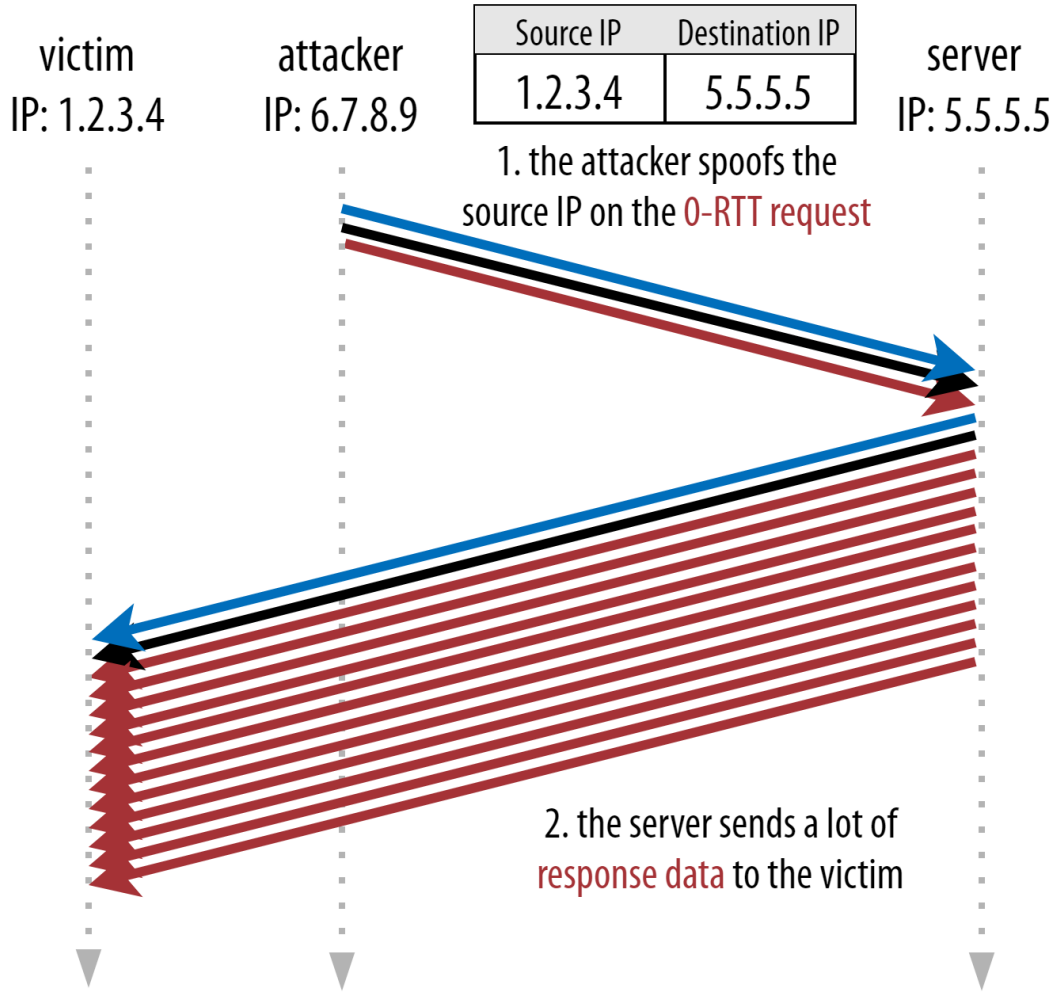
https://asnokaze.hatenablog.com/entry/2019/04/22/000927

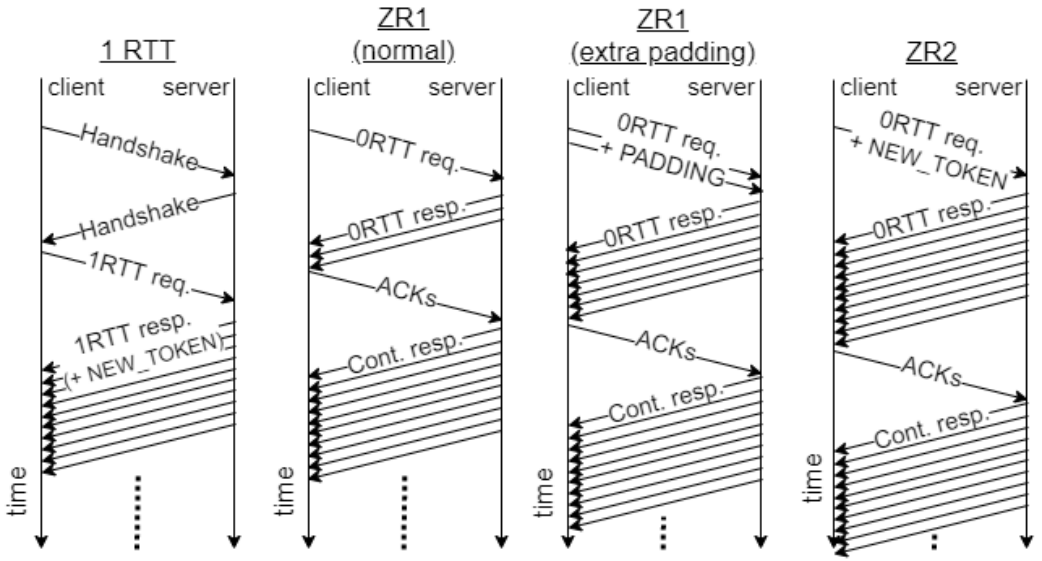
Combined Transport and Crypto handshake



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

Amplification prevention/mitigation



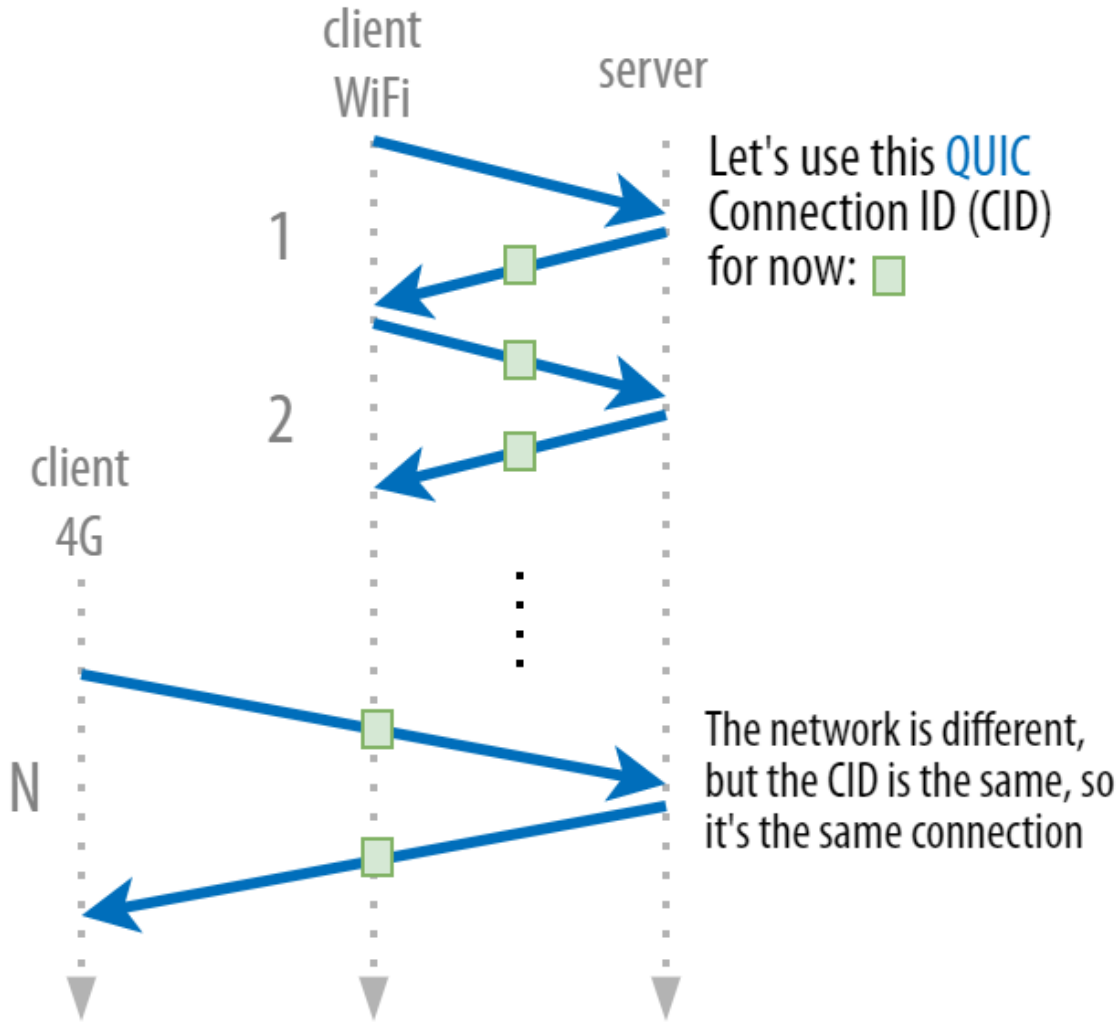


Other issues

- Replay attacks
- STEK rotation
- → not QUIC specific, also for TCP+TLS resumption/0-RTT

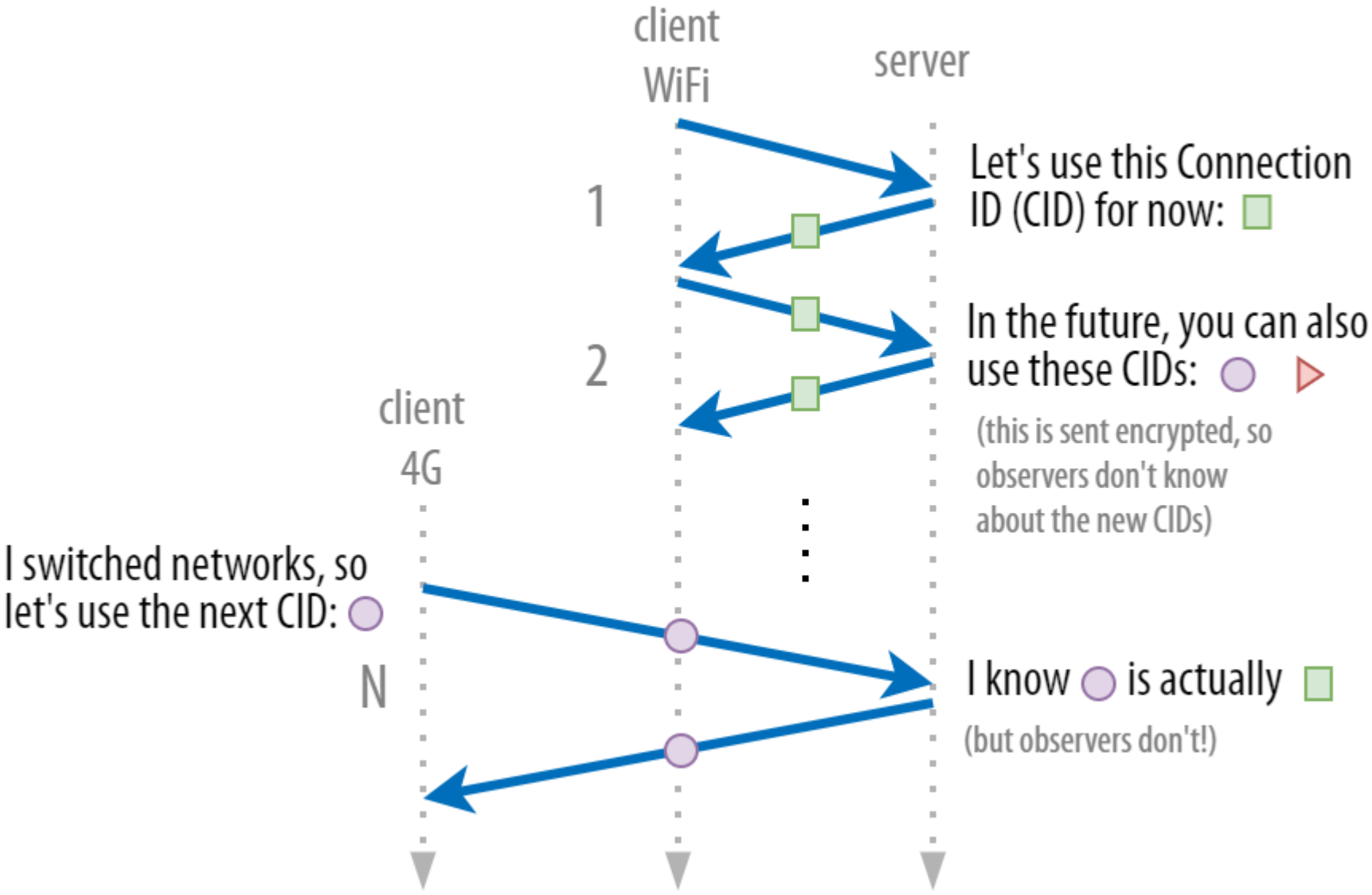
https://qlog.edm.uhasselt.be/epiq
https://www.smashingmagazine.com/2021/08/http3-core-concepts-part1

Connection migration is a privacy nightmare



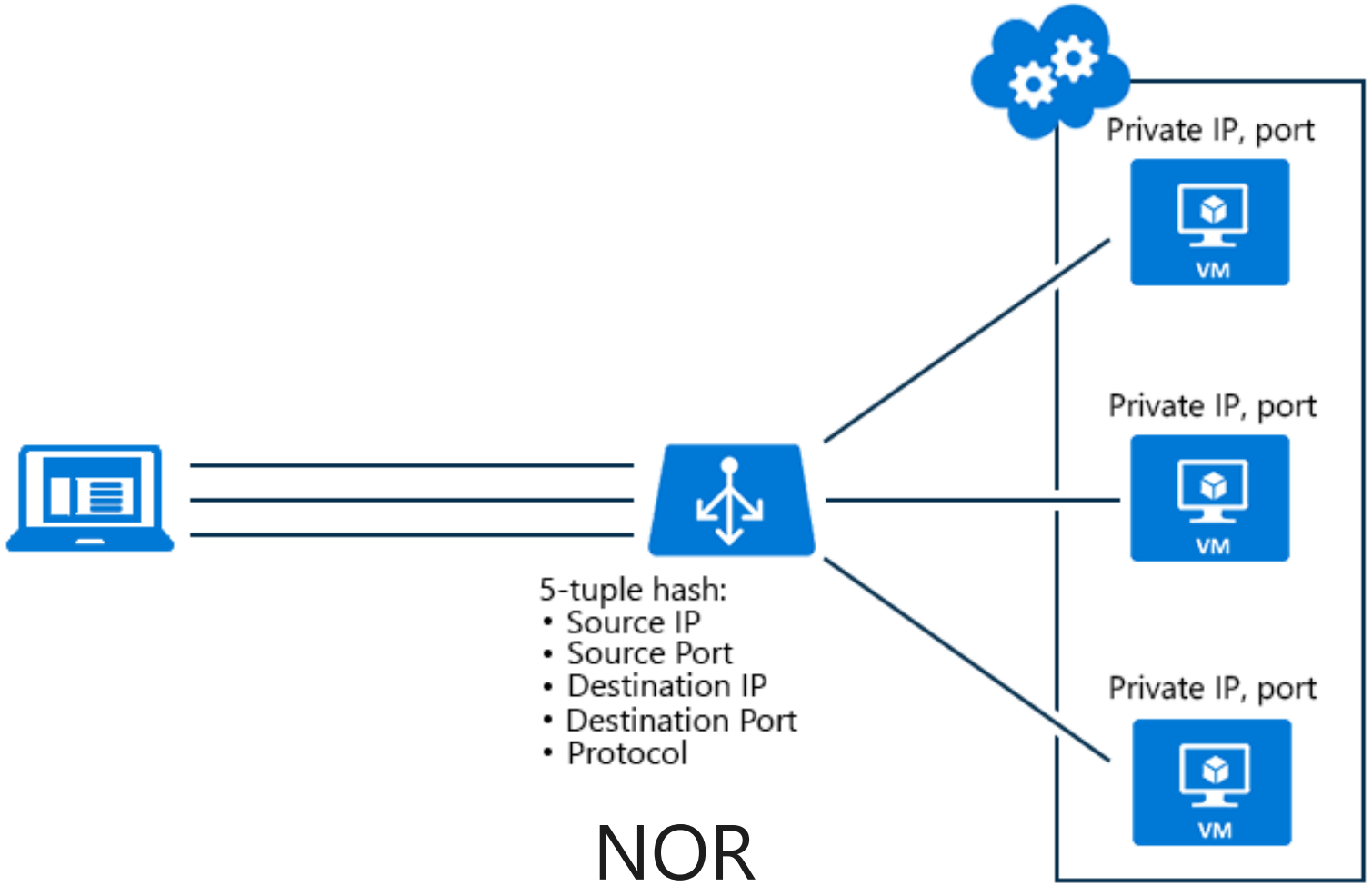
https://www.smashingmagazine.com/2021/08/http3-core-concepts-part1/

Linkability prevention with migration



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

Linkability prevention is deployment nightmare

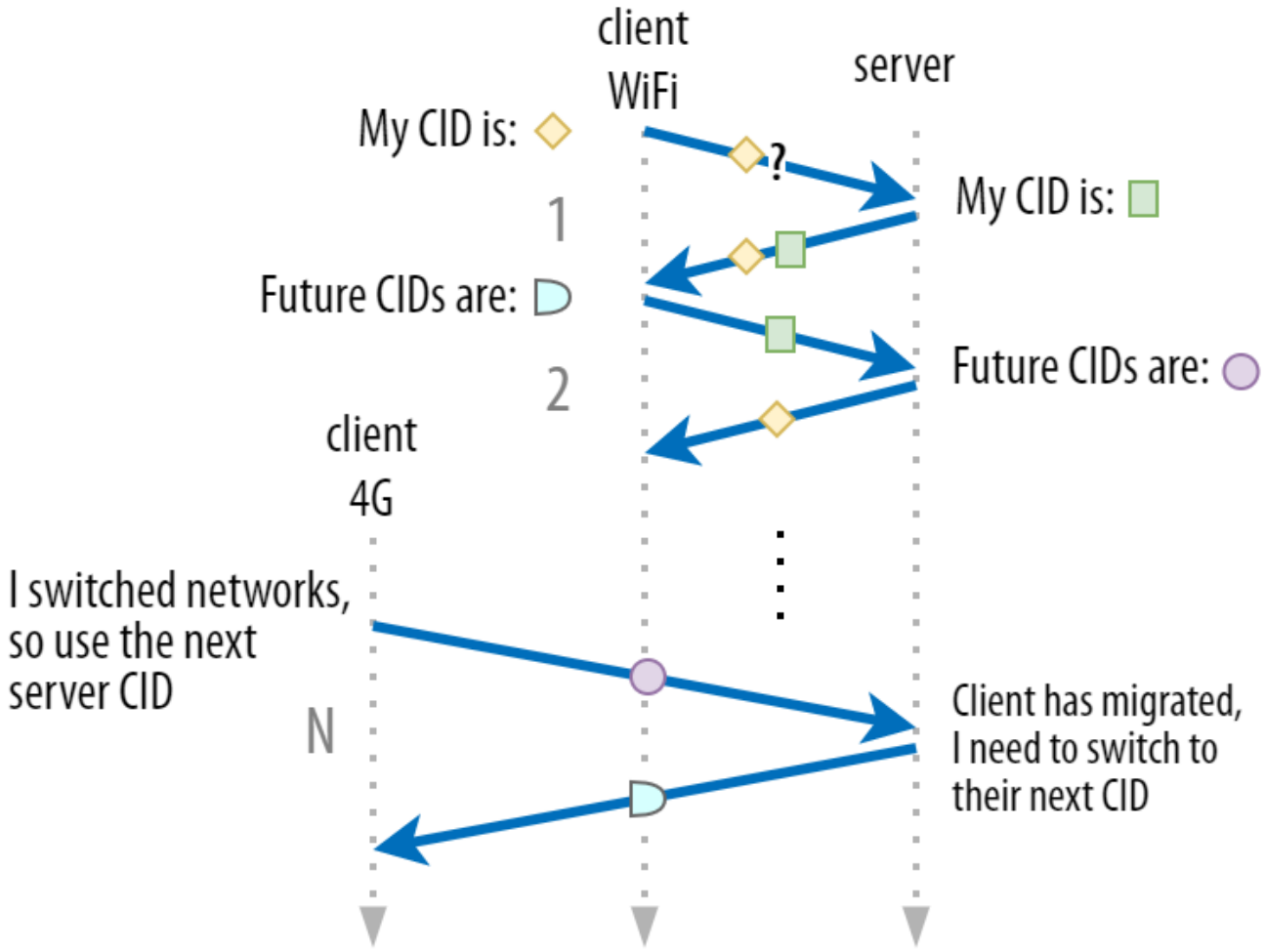


NOR

Connection ID

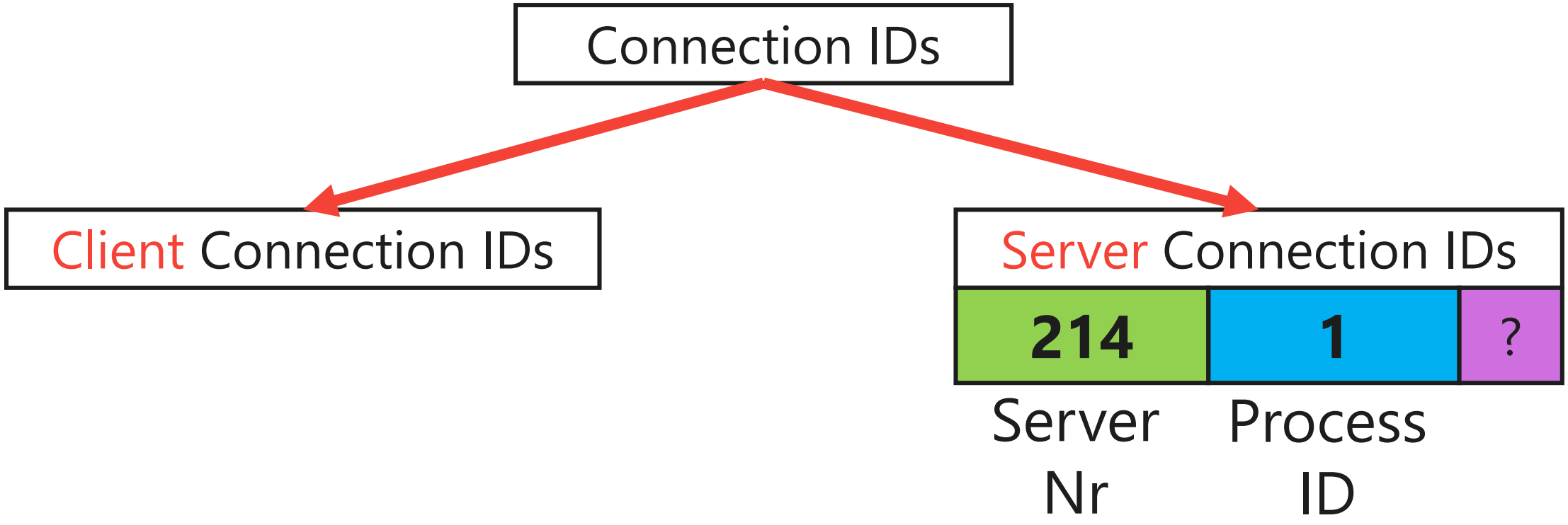
https://tools.ietf.org/html/draft-duke-quic-load-balancers

Splitting CIDs for routability



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

Splitting CIDs for routability

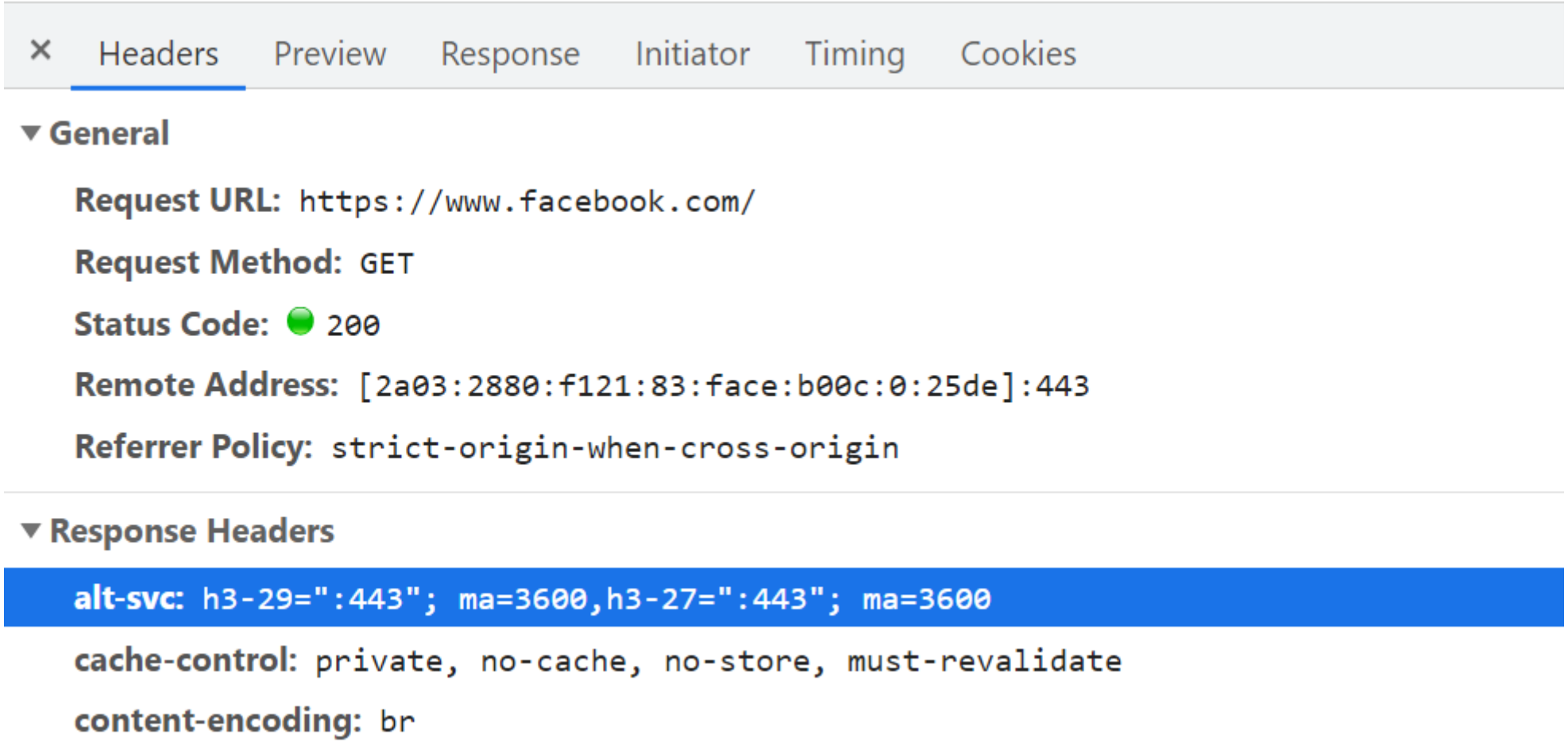


Need to use standard/configured/negotiated format for:

- Load balancers
- Potentially also firewalls!

https://www.youtube.com/watch?v=pq_xk_Pecu4

Open that UDP:443



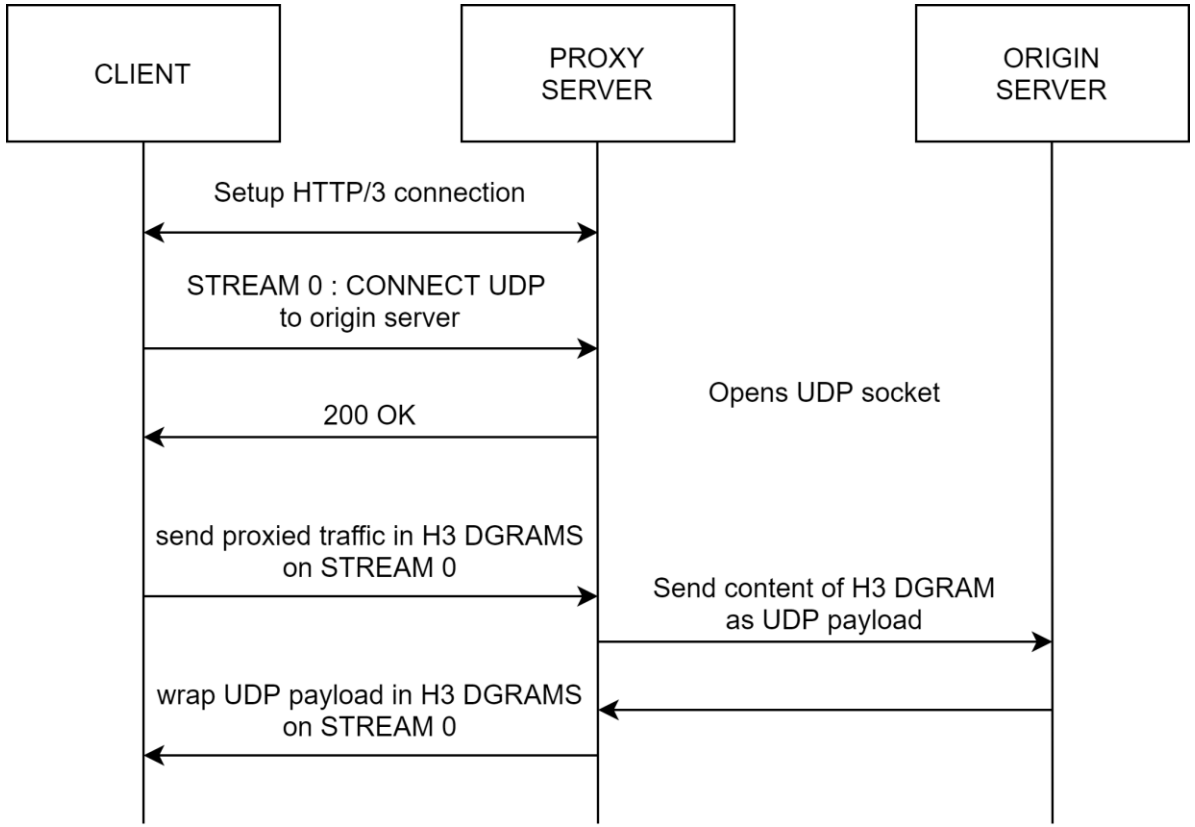
Tunneling/Proxying stuff over QUIC

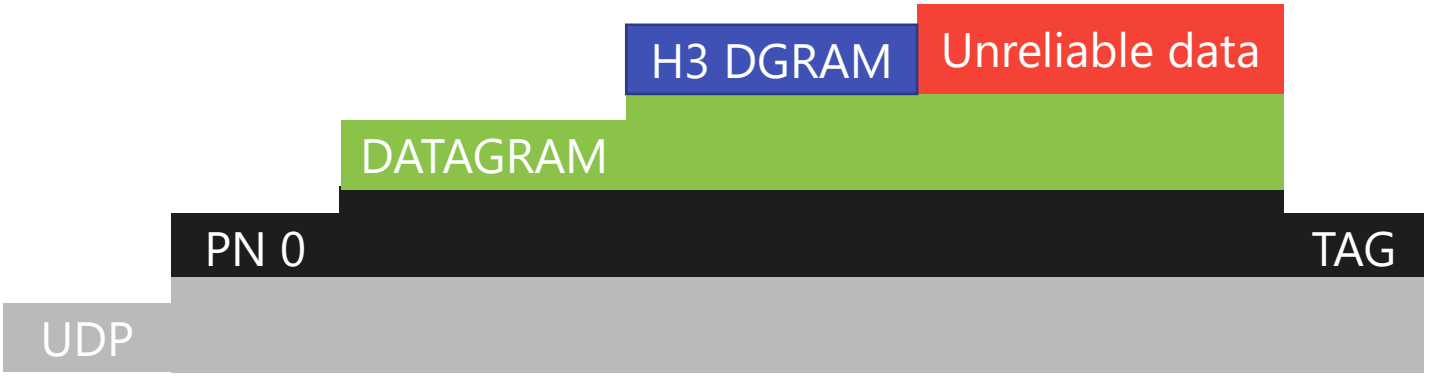
Lots of stuff proposed:

- DNS over QUIC / H3
- WebTransport
- SSH over QUIC
- …

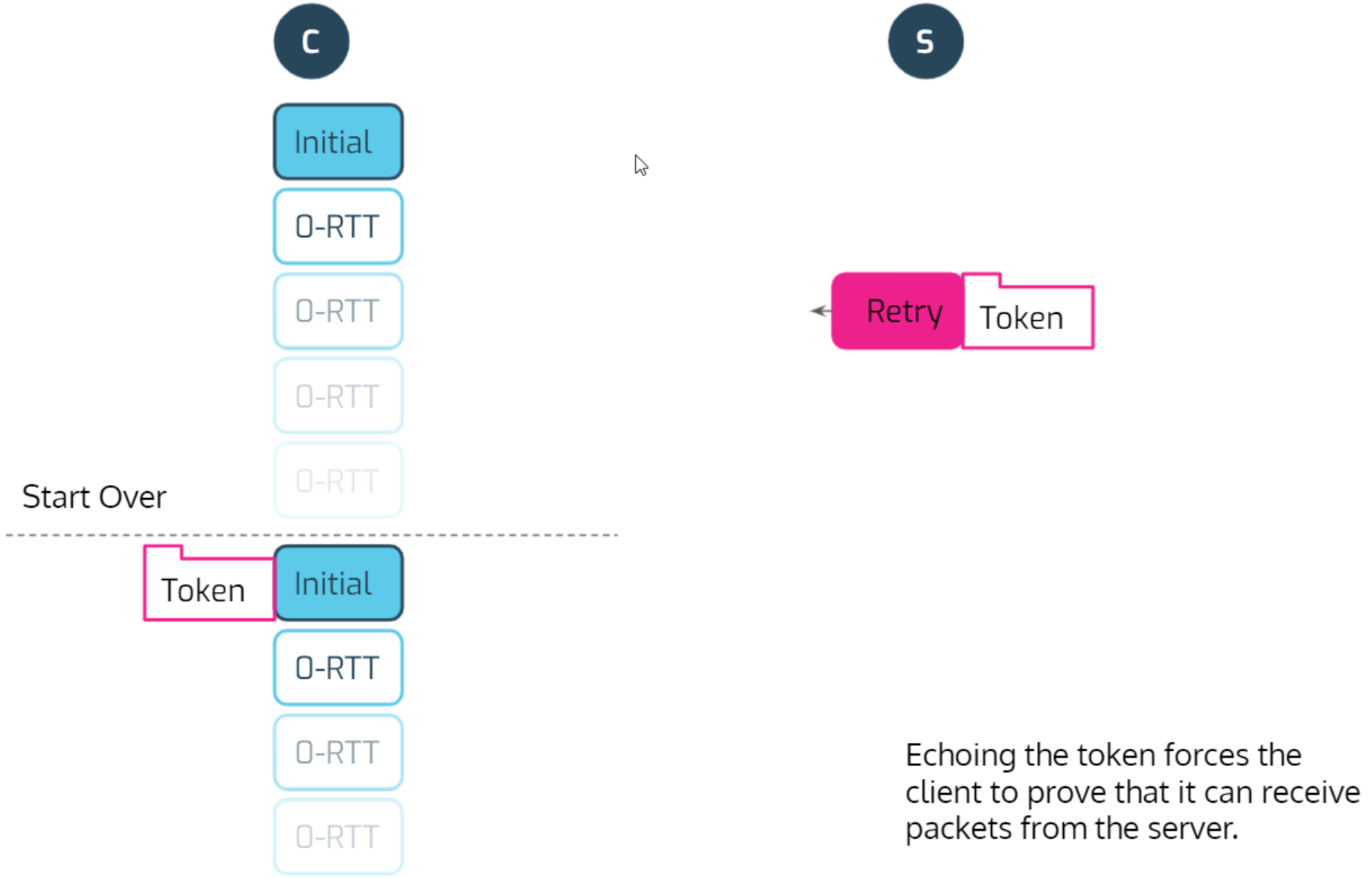
Block all or nothing

- H3 falls back to H2 over TCP
- Other things might not be so lucky…





TCP SYN Cookies ~= QUIC RETRY token



Echoing the token forces the client to prove that it can receive packets from the server.

https://docs.google.com/presentation/d/1OASDYIJlgSFg6hRkUjqdKfYTK1ZUk5VMGP3Iv2zQCI8

Prevent TLS interceptors from messing with QUIC

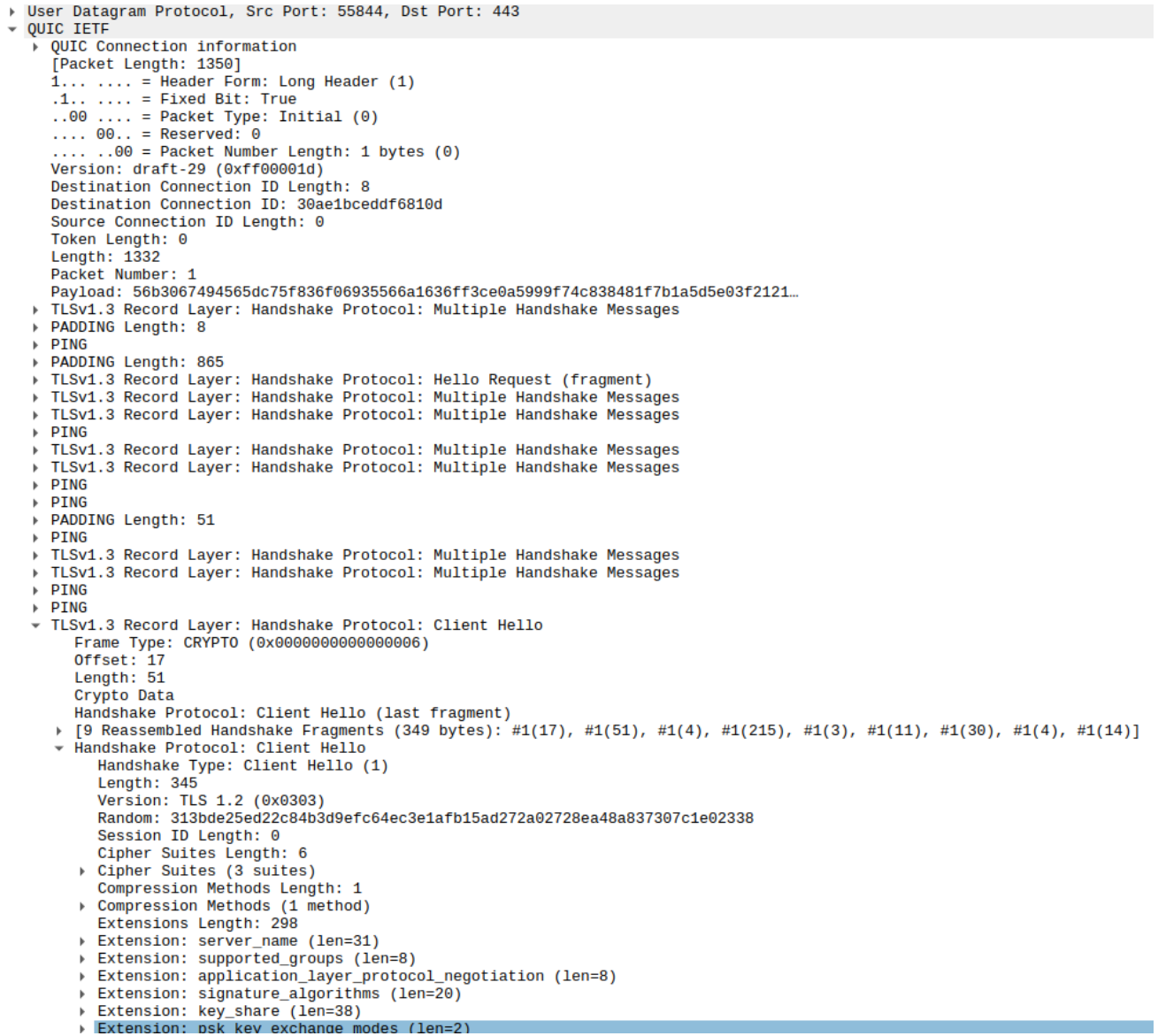
dschinazi 9:17 PM

QUIC uses the local store but it fails the handshake if the root CA is not in the default set. This was done by policy to prevent antivirus software from MITM QUIC so we can keep evolving QUIC.

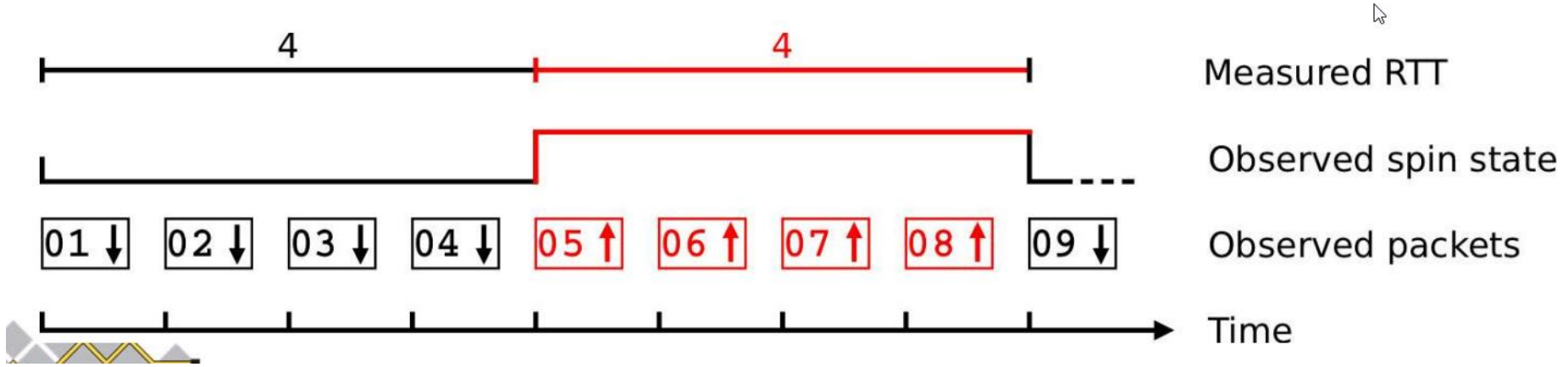
```
35  options = webdriver.ChromeOptions()
36  options.gpu = False
37  options.headless = True
38  options.binary_location = "/usr/bin/google-chrome-beta"
39  options.add_argument("--no-sandbox")
40  options.add_argument("--enable-quic")
41  options.add_argument("--quic-version=h3-29")
42  options.add_argument("--origin-to-force-quic-on=" + server)
43  options.add_argument("--log-net-log=/logs/chrome.json")
44  options.add_argument("--net-log-capture-mode=IncludeSensitive")
45  options.add_argument("--ignore-certificate-errors-spki-list=" + get_args().certhash)
```

https://www.chromium.org/quic/playing-with-quic
https://github.com/marten-seemann/chrome-quic-interop-runner/blob/master/run.py

Prevent TLS interceptors from messing with QUIC

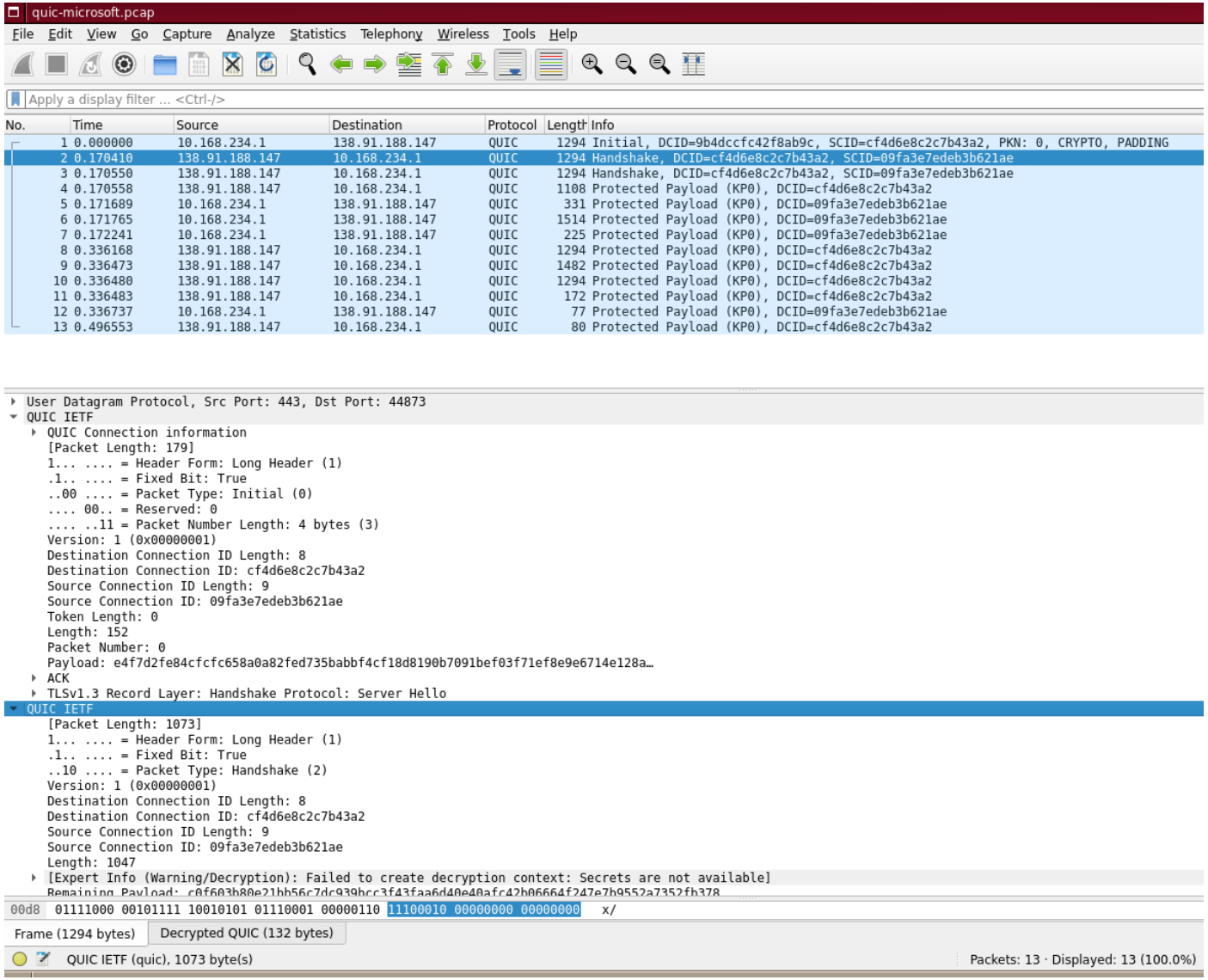


QUIC observability / debuggability



https://datatracker.ietf.org/doc/html/draft-ietf-quic-applicability
https://datatracker.ietf.org/doc/html/draft-ietf-quic-manageability
https://www.ietfjournal.org/enabling-internet-measurement-with-the-quic-spin-bit

QUIC observability / debuggability



SSLKEYLOGFILE

https://www.bortzmeyer.org/quic-demo.html

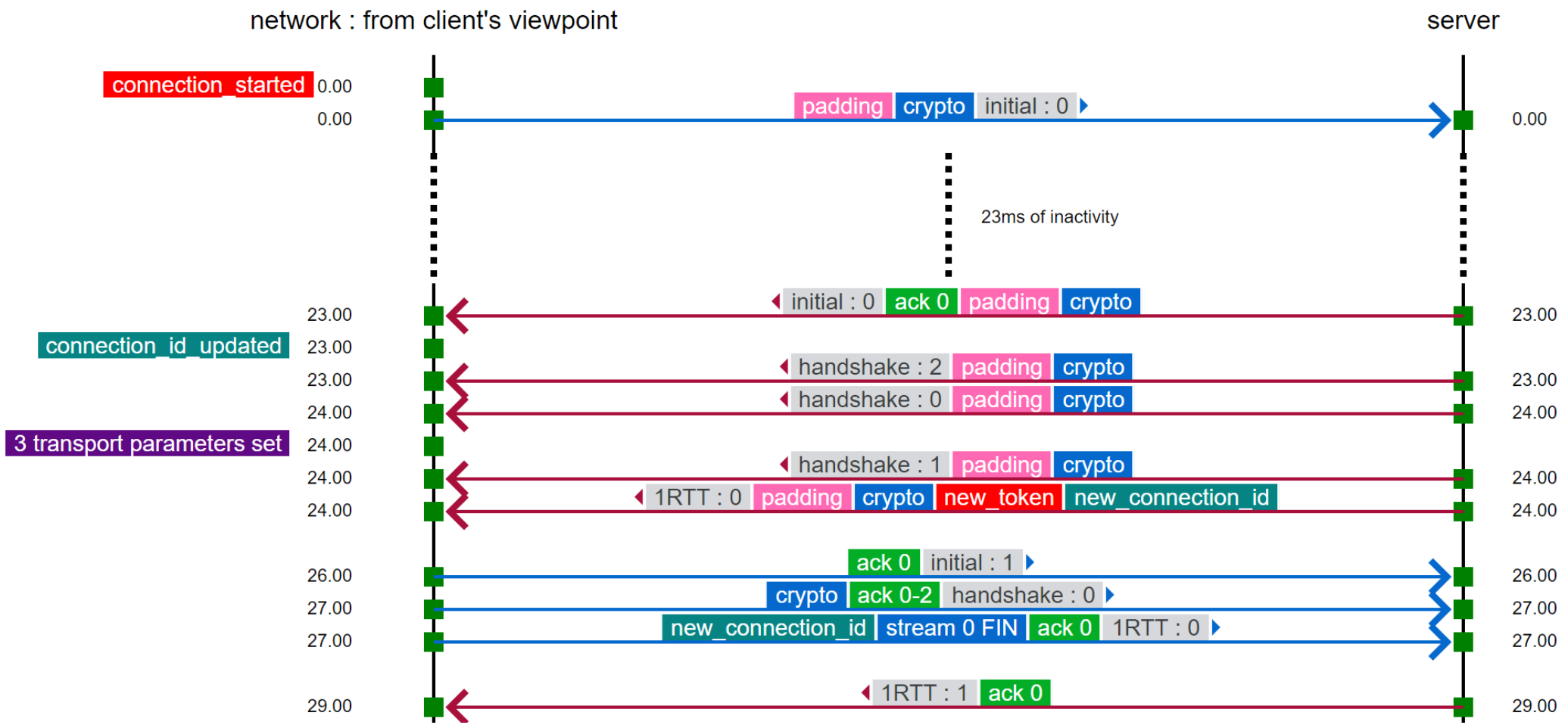
[qlog] structured endpoint logging

```
{
"event_fields": [
        "time","group_id", "category",  "event",           "data"
    ],
    "events": [
        [1553986553579, 0, "transport", "packet_received",  {"header": {}}],
        [1553986553580, 0, "recovery",  "metrics_updated",  {"smoothed_rtt": 85}],
        [1553986553588, 1, "http",      "frame_parsed",     {"frame_type": "DATA"}],
        [1553986553598, 0, "transport", "packet_sent",      {"header": {}}]
    ]
}
```

get data from implementations directly

https://github.com/quicwg/qlog

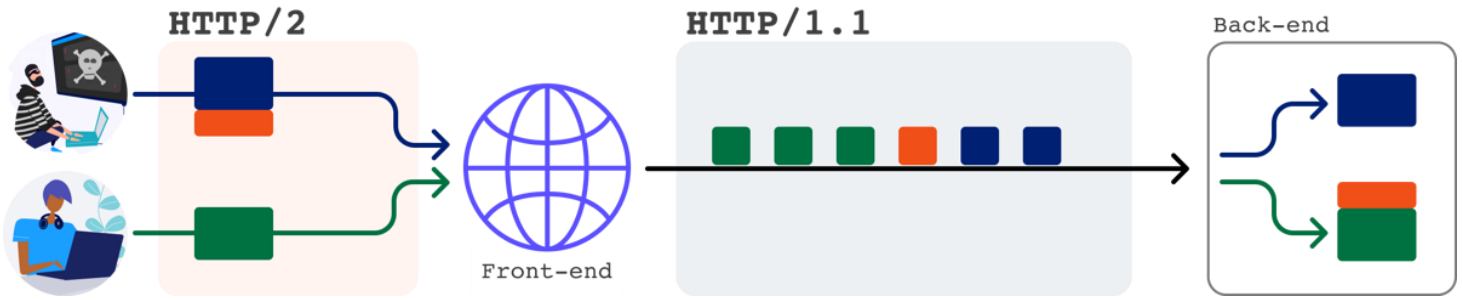
<qvis> QUIC and HTTP/3 tools

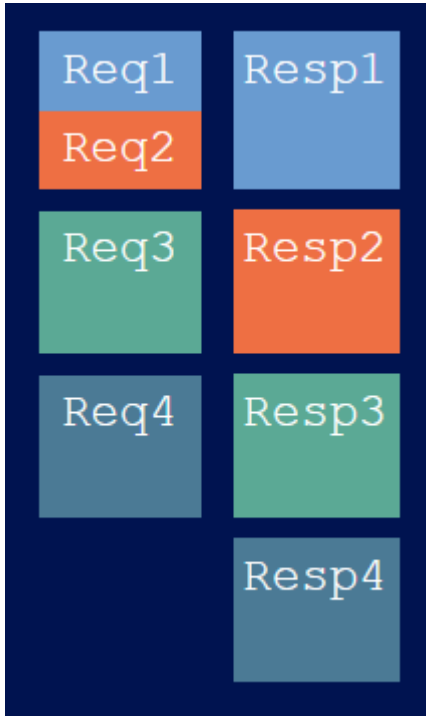


https://qvis.quictools.info

https://github.com/quiclog/qvis

HTTP/2 request smuggling






```
POST /n HTTP/1.1
Host: www.netflix.com
Content-Length: 4

abcdGET /n HTTP/1.1
Host: 02.rs?x.netflix.com
Foo: bar
```

https://portswigger.net/research/http2

Image sources

1. See bottom right of most slides
2. https://github.com/rmarx/http3-for-webdevs = collection of free to re-use diagrams