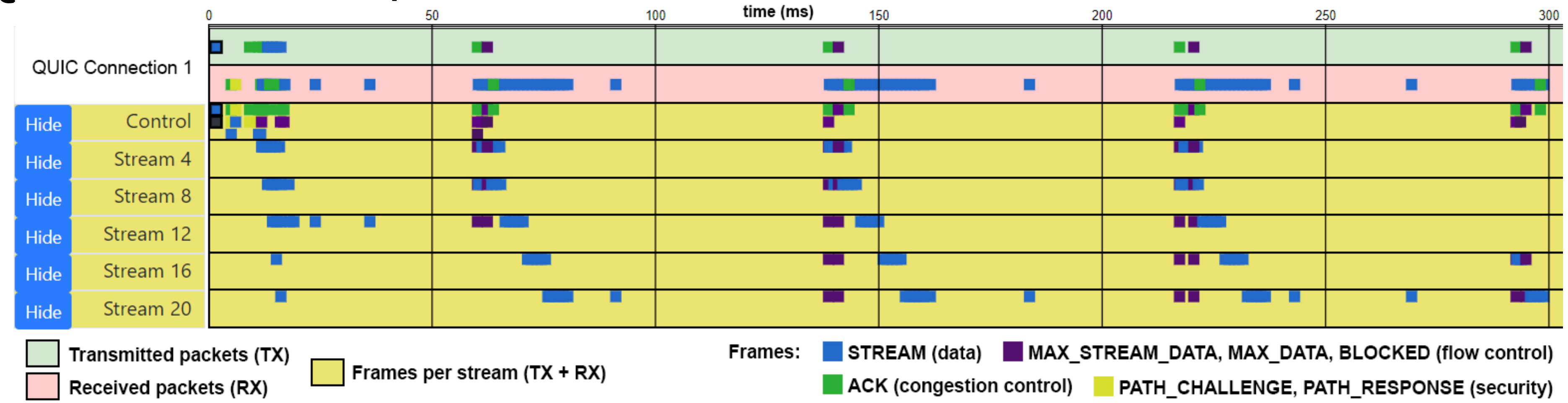
An update on qlog

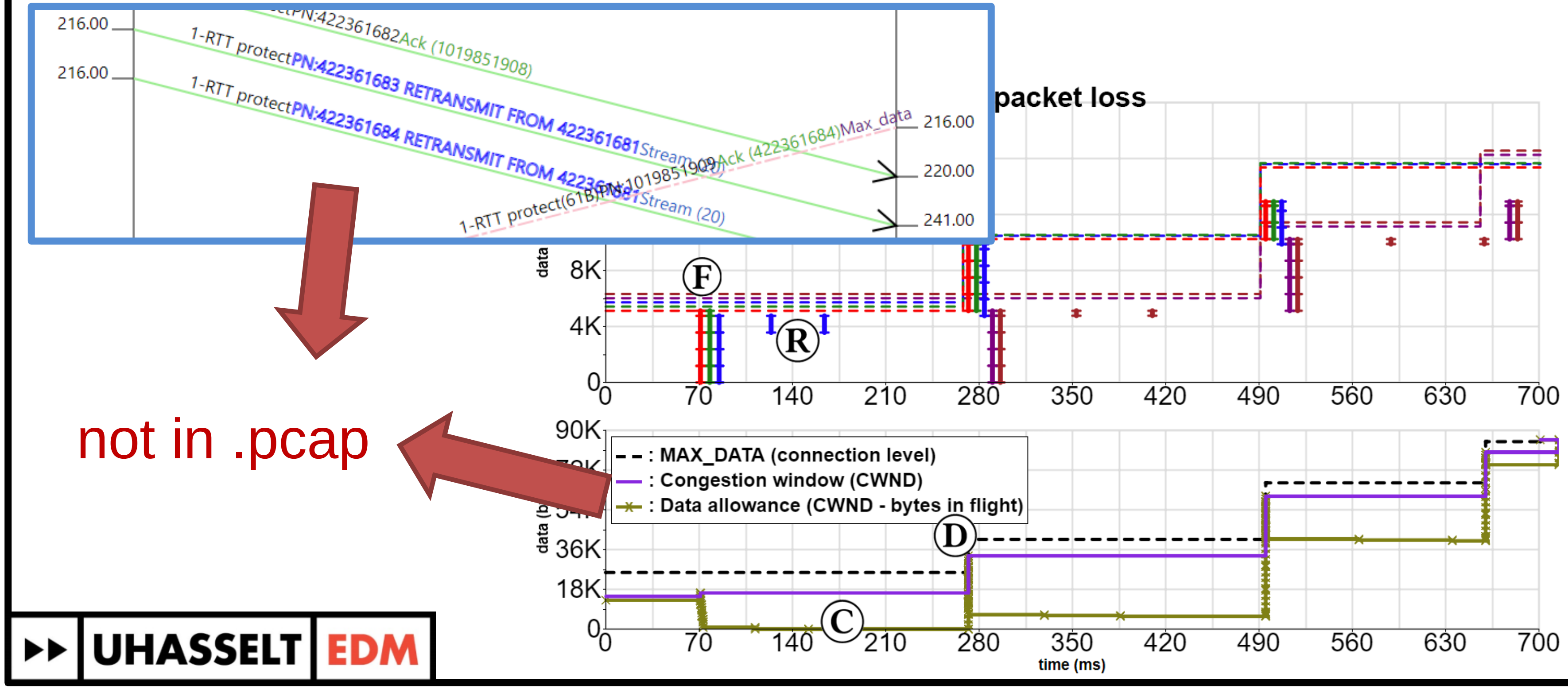
Robin Marx - @programmingart
(Hasselt University – Belgium)

UHASSELT EDM

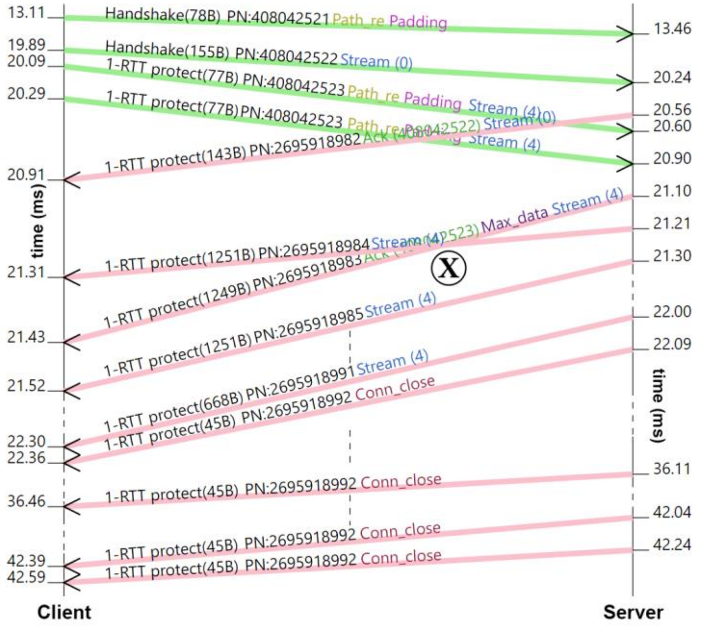
IETF 105 – Montreal – 25 July 2019

Motivation: QUIC and HTTP/3 tools





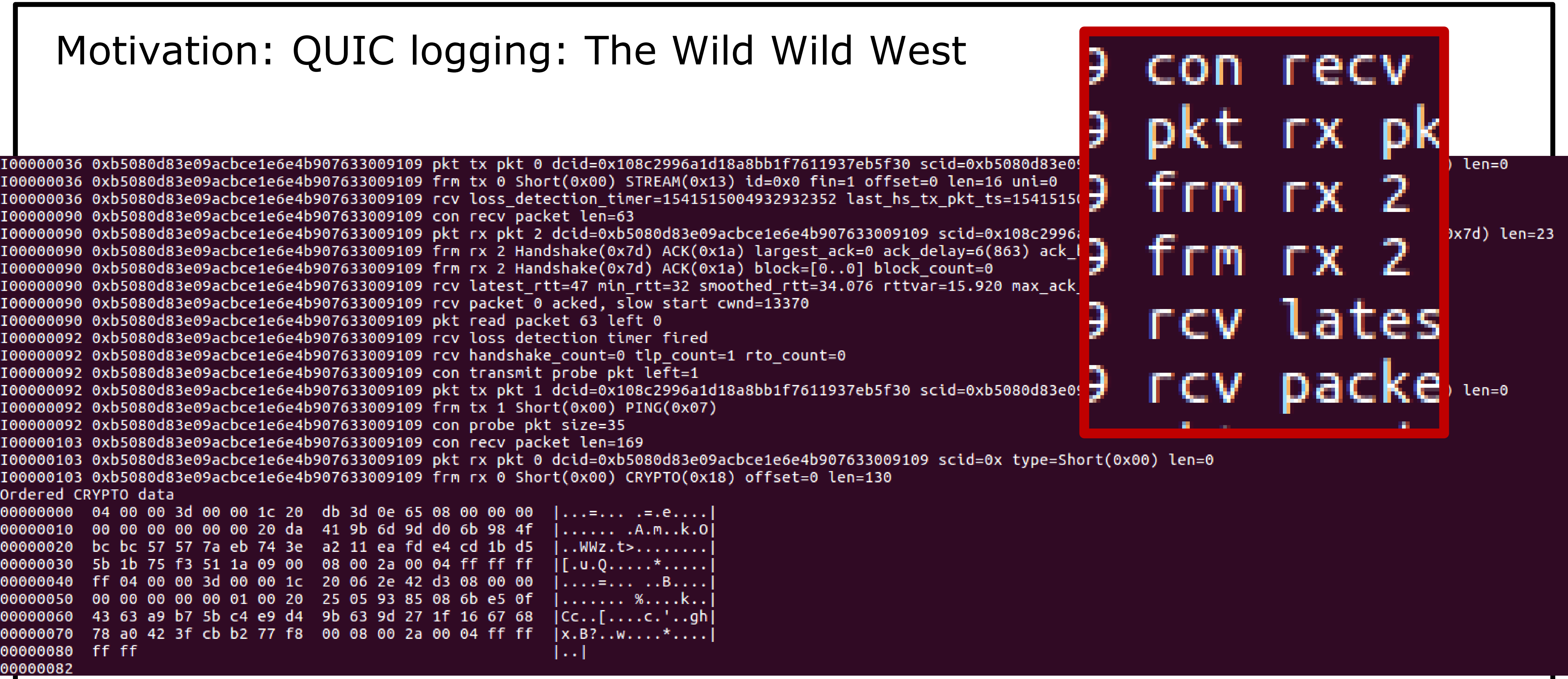
not in .pcap


13.11
19.89
20.09
20.29
20.91
21.31
21.43
21.52
22.30
22.36
36.46
42.39
42.59
Handshake(78B) PN:408042521 Path_re Padding
Handshake(155B) PN:408042522 Stream (0)
1-RTT protect(77B) PN:408042523 Path_re Padding Stream (4)(0)
1-RTT protect(77B) PN:408042523 Path_re Padding Stream (4)
1-RTT protect(143B) PN:2695918982 Ack (408042522)
1-RTT protect(1251B) PN:2695918984 Stream (4)
(408042523) Max_data Stream (4)
1-RTT protect(1249B) PN:2695918983 Ack
X
1-RTT protect(1251B) PN:2695918985 Stream (4)
1-RTT protect(668B) PN:2695918991 Stream (4)
1-RTT protect(45B) PN:2695918992 Conn_close
1-RTT protect(45B) PN:2695918992 Conn_close
1-RTT protect(45B) PN:2695918992 Conn_close
1-RTT protect(45B) PN:2695918992 Conn_close
13.46
20.24
20.56
20.60
20.90
21.10
21.21
21.30
22.00
22.09
36.11
42.04
42.24
time (ms)
Client
Server

UHASSELT EDM

https://quic.edm.uhasselt.be

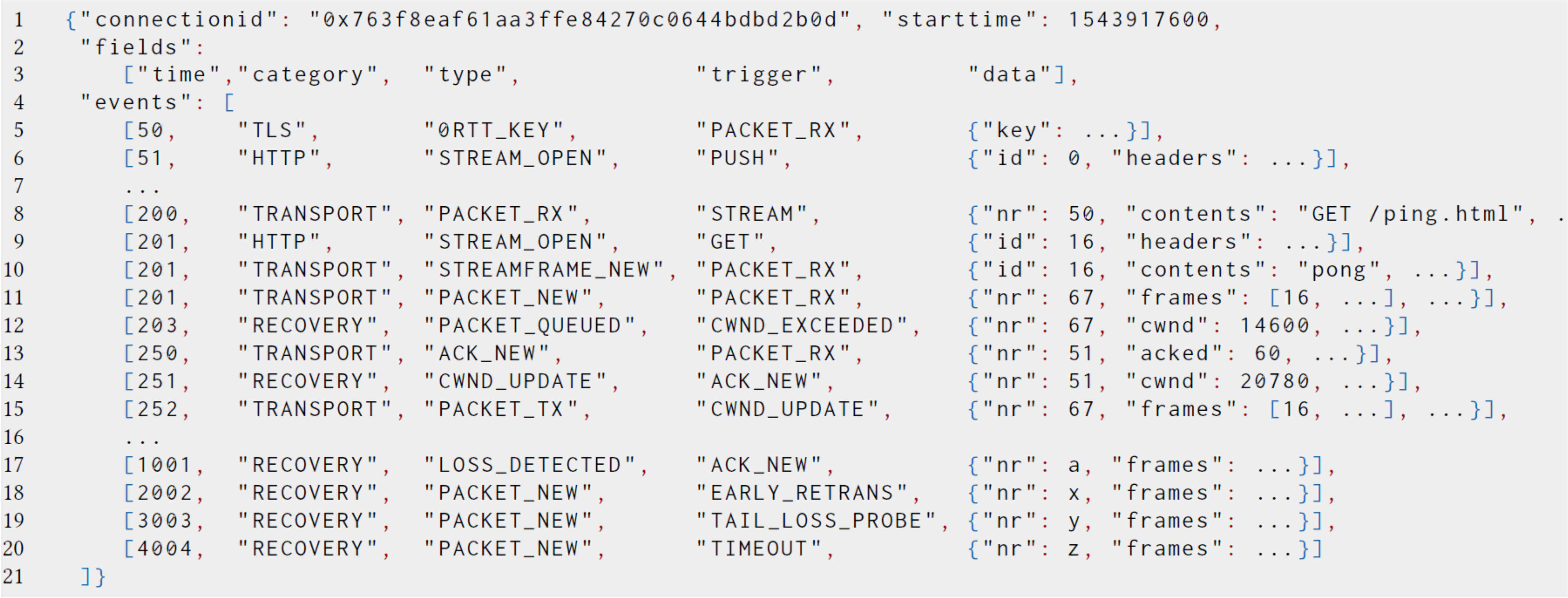
Motivation: QUIC logging: The Wild Wild West



```
con recv
pkt rx pk
frm rx 2
frm rx 2
rcv lates
rcv packe
```

```
I00000036 0xb5080d83e09acbce1e6e4b907633009109 pkt tx pkt 0 dcid=0x108c2996a1d18a8bb1f7611937eb5f30 scid=0xb5080d83e09 ) len=0
I00000036 0xb5080d83e09acbce1e6e4b907633009109 frm tx 0 Short(0x00) STREAM(0x13) id=0x0 fin=1 offset=0 len=16 uni=0
I00000036 0xb5080d83e09acbce1e6e4b907633009109 rcv loss_detection_timer=1541515004932932352 last_hs_tx_pkt_ts=1541515
I00000090 0xb5080d83e09acbce1e6e4b907633009109 con recv packet len=63
I00000090 0xb5080d83e09acbce1e6e4b907633009109 pkt rx pkt 2 dcid=0xb5080d83e09acbce1e6e4b907633009109 scid=0x108c2996a 0x7d) len=23
I00000090 0xb5080d83e09acbce1e6e4b907633009109 frm rx 2 Handshake(0x7d) ACK(0x1a) largest_ack=0 ack_delay=6(863) ack_
I00000090 0xb5080d83e09acbce1e6e4b907633009109 frm rx 2 Handshake(0x7d) ACK(0x1a) block=[0..0] block_count=0
I00000090 0xb5080d83e09acbce1e6e4b907633009109 rcv latest_rtt=47 min_rtt=32 smoothed_rtt=34.076 rttvar=15.920 max_ack_
I00000090 0xb5080d83e09acbce1e6e4b907633009109 rcv packet 0 acked, slow start cwnd=13370
I00000090 0xb5080d83e09acbce1e6e4b907633009109 pkt read packet 63 left 0
I00000092 0xb5080d83e09acbce1e6e4b907633009109 rcv loss detection timer fired
I00000092 0xb5080d83e09acbce1e6e4b907633009109 rcv handshake_count=0 tlp_count=1 rto_count=0
I00000092 0xb5080d83e09acbce1e6e4b907633009109 con transmit probe pkt left=1
I00000092 0xb5080d83e09acbce1e6e4b907633009109 pkt tx pkt 1 dcid=0x108c2996a1d18a8bb1f7611937eb5f30 scid=0xb5080d83e09 ) len=0
I00000092 0xb5080d83e09acbce1e6e4b907633009109 frm tx 1 Short(0x00) PING(0x07)
I00000092 0xb5080d83e09acbce1e6e4b907633009109 con probe pkt size=35
I00000103 0xb5080d83e09acbce1e6e4b907633009109 con recv packet len=169
I00000103 0xb5080d83e09acbce1e6e4b907633009109 pkt rx pkt 0 dcid=0xb5080d83e09acbce1e6e4b907633009109 scid=0x type=Short(0x00) len=0
I00000103 0xb5080d83e09acbce1e6e4b907633009109 frm rx 0 Short(0x00) CRYPTO(0x18) offset=0 len=130
Ordered CRYPTO data
00000000  04 00 00 3d 00 00 1c 20  db 3d 0e 65 08 00 00 00  |...=... .=.e....|
00000010  00 00 00 00 00 00 20 da  41 9b 6d 9d d0 6b 98 4f  |...... .A.m..k.O|
00000020  bc bc 57 57 7a eb 74 3e  a2 11 ea fd e4 cd 1b d5  |..WWz.t>........|
00000030  5b 1b 75 f3 51 1a 09 00  08 00 2a 00 04 ff ff ff  |[.u.Q.....*.....|
00000040  ff 04 00 00 3d 00 00 1c  20 06 2e 42 d3 08 00 00  |....=... ..B....|
00000050  00 00 00 00 00 01 00 20  25 05 93 85 08 6b e5 0f  |....... %....k..|
00000060  43 63 a9 b7 5b c4 e9 d4  9b 63 9d 27 1f 16 67 68  |Cc..[....c.'..gh|
00000070  78 a0 42 3f cb b2 77 f8  00 08 00 2a 00 04 ff ff  |x.B?..w....*....|
00000080  ff ff                                             |..|
00000082
```

Our proposal: qlog

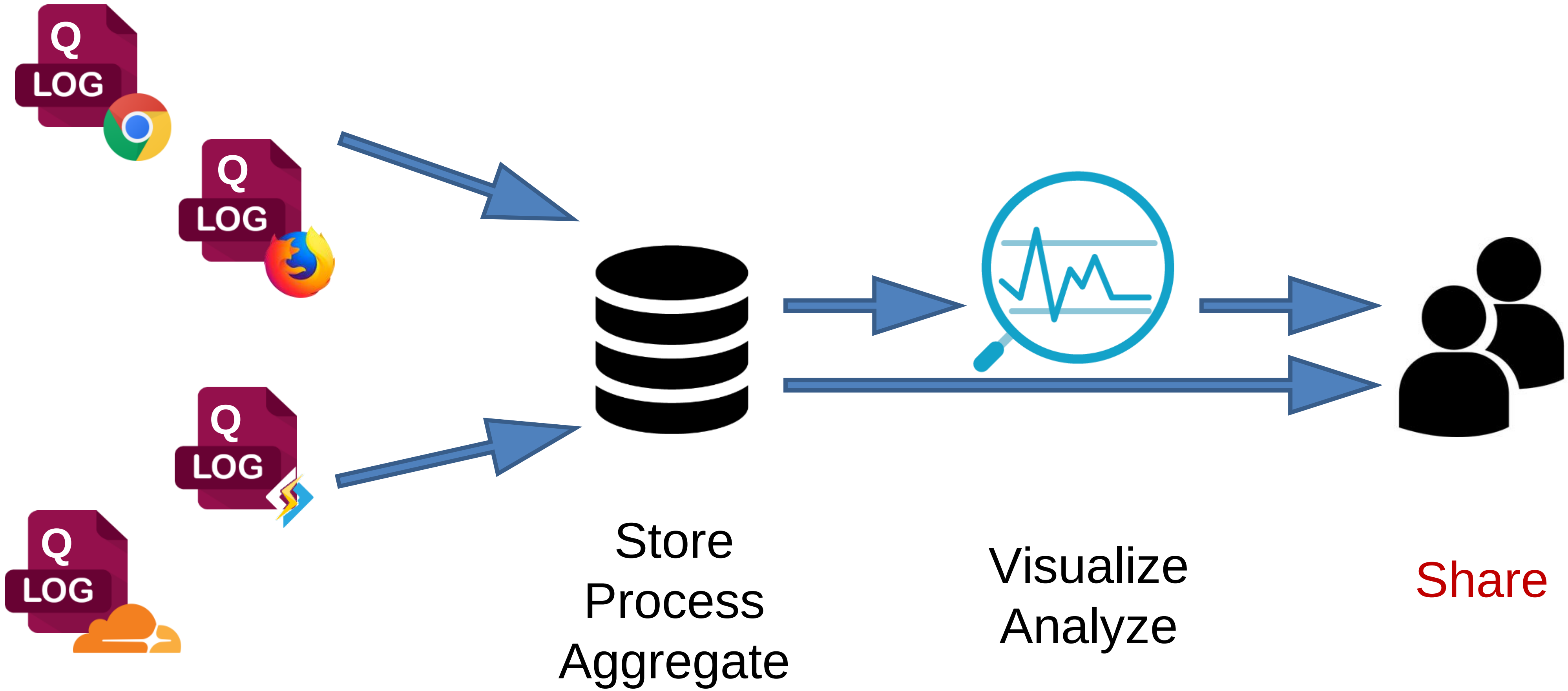


```
{"connectionid": "0x763f8eaf61aa3ffe84270c0644bdbd2b0d", "starttime": 1543917600,
 "fields":
    ["time","category",  "type",            "trigger",         "data"],
 "events": [
    [50,    "TLS",       "0RTT_KEY",        "PACKET_RX",       {"key": ...}],
    [51,    "HTTP",      "STREAM_OPEN",     "PUSH",            {"id": 0, "headers": ...}],
    ...
    [200,   "TRANSPORT", "PACKET_RX",       "STREAM",          {"nr": 50, "contents": "GET /ping.html", .
    [201,   "HTTP",      "STREAM_OPEN",     "GET",             {"id": 16, "headers": ...}],
    [201,   "TRANSPORT", "STREAMFRAME_NEW", "PACKET_RX",       {"id": 16, "contents": "pong", ...}],
    [201,   "TRANSPORT", "PACKET_NEW",      "PACKET_RX",       {"nr": 67, "frames": [16, ...], ...}],
    [203,   "RECOVERY",  "PACKET_QUEUED",   "CWND_EXCEEDED",   {"nr": 67, "cwnd": 14600, ...}],
    [250,   "TRANSPORT", "ACK_NEW",         "PACKET_RX",       {"nr": 51, "acked": 60, ...}],
    [251,   "RECOVERY",  "CWND_UPDATE",     "ACK_NEW",         {"nr": 51, "cwnd": 20780, ...}],
    [252,   "TRANSPORT", "PACKET_TX",       "CWND_UPDATE",     {"nr": 67, "frames": [16, ...], ...}],
    ...
    [1001,  "RECOVERY",  "LOSS_DETECTED",   "ACK_NEW",         {"nr": a, "frames": ...}],
    [2002,  "RECOVERY",  "PACKET_NEW",      "EARLY_RETRANS",   {"nr": x, "frames": ...}],
    [3003,  "RECOVERY",  "PACKET_NEW",      "TAIL_LOSS_PROBE", {"nr": y, "frames": ...}],
    [4004,  "RECOVERY",  "PACKET_NEW",      "TIMEOUT",         {"nr": z, "frames": ...}]
 ]}
```

JSON:

- Easy to use in web-based tools (and most programming languages)
- Human-readable

Standardized QUIC endpoint logging format



Store
Process
Aggregate

Visualize
Analyze

Share

Standardized general purpose endpoint logging format?

- Why just for QUIC and HTTP/3?
  - TCP endpoint states
  - RTP / WebRTC / DTLS
  - Anything really…
- Robin’s logging best practices:
  - Event-based
  - Multiple vantage points (endpoints + in-network)
  - Flexible (support custom data)
  - Accessible (where and how to obtain logs)
  - Privacy and Security

Discussed at IETF 104

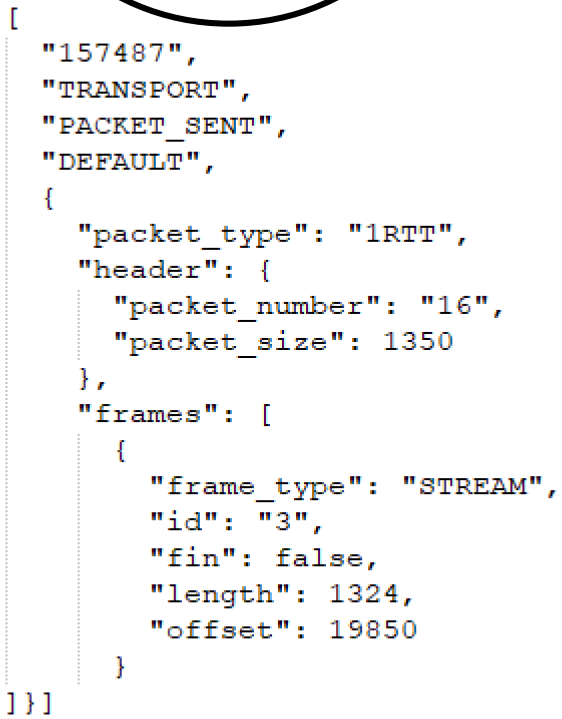
- Interesting, but too early for BOF / its own working group
- Use QUIC + H3 as incubator / concrete use case

https://www.youtube.com/embed/LiNLz1QuT0s?start=3199
https://datatracker.ietf.org/meeting/104/materials/slides-104-tsvarea-logging-tooling-and-debugging-for-modern-network-protocols-robin-marx-00

Standardized general purpose endpoint logging format, in 2 parts!

High-level schema

QUIC + HTTP/3 event definitions

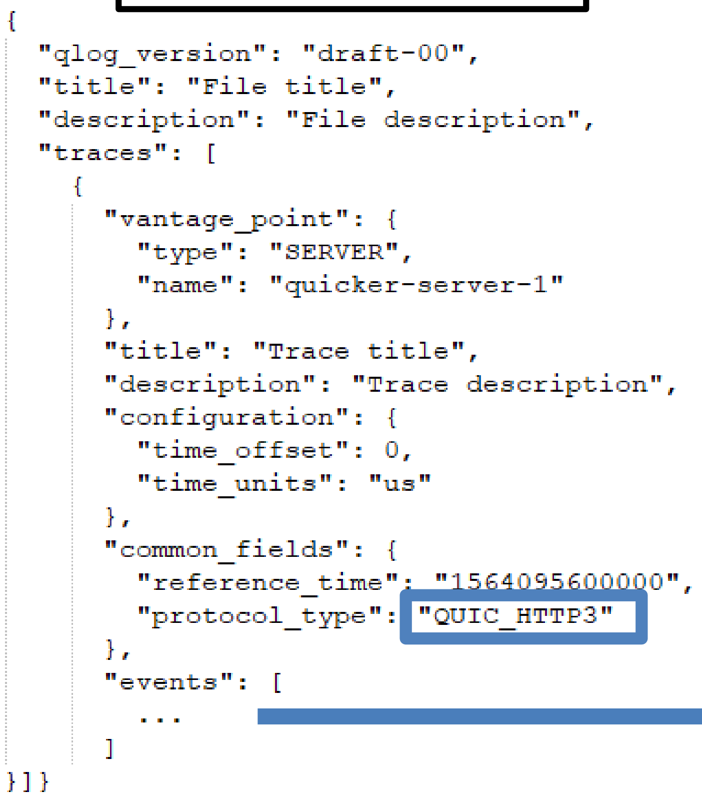


```
[
  "157487",
  "TRANSPORT",
  "PACKET_SENT",
  "DEFAULT",
  {
    "packet_type": "1RTT",
    "header": {
      "packet_number": "16",
      "packet_size": 1350
    },
    "frames": [
      {
        "frame_type": "STREAM",
        "id": "3",
        "fin": false,
        "length": 1324,
        "offset": 19850
      }
]}]
```

https://tools.ietf.org/html/draft-marx-qlog-main-schema
https://tools.ietf.org/html/draft-marx-qlog-event-definitions-quic-h3

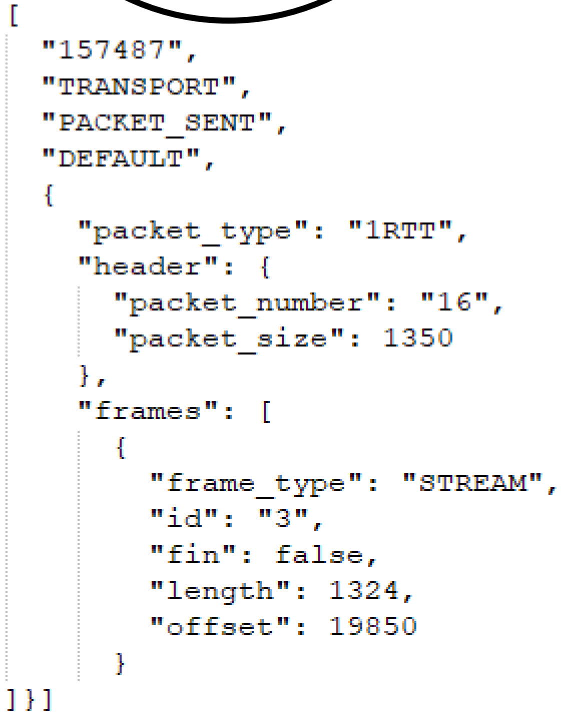
Standardized general purpose endpoint logging format, in 2 parts!

High-level schema



```
{
  "qlog_version": "draft-00",
  "title": "File title",
  "description": "File description",
  "traces": [
    {
      "vantage_point": {
        "type": "SERVER",
        "name": "quicker-server-1"
      },
      "title": "Trace title",
      "description": "Trace description",
      "configuration": {
        "time_offset": 0,
        "time_units": "us"
      },
      "common_fields": {
        "reference_time": "1564095600000",
        "protocol_type": "QUIC_HTTP3"
      },
      "events": [
        ...
      ]
}]}
```

QUIC + HTTP/3 event definitions

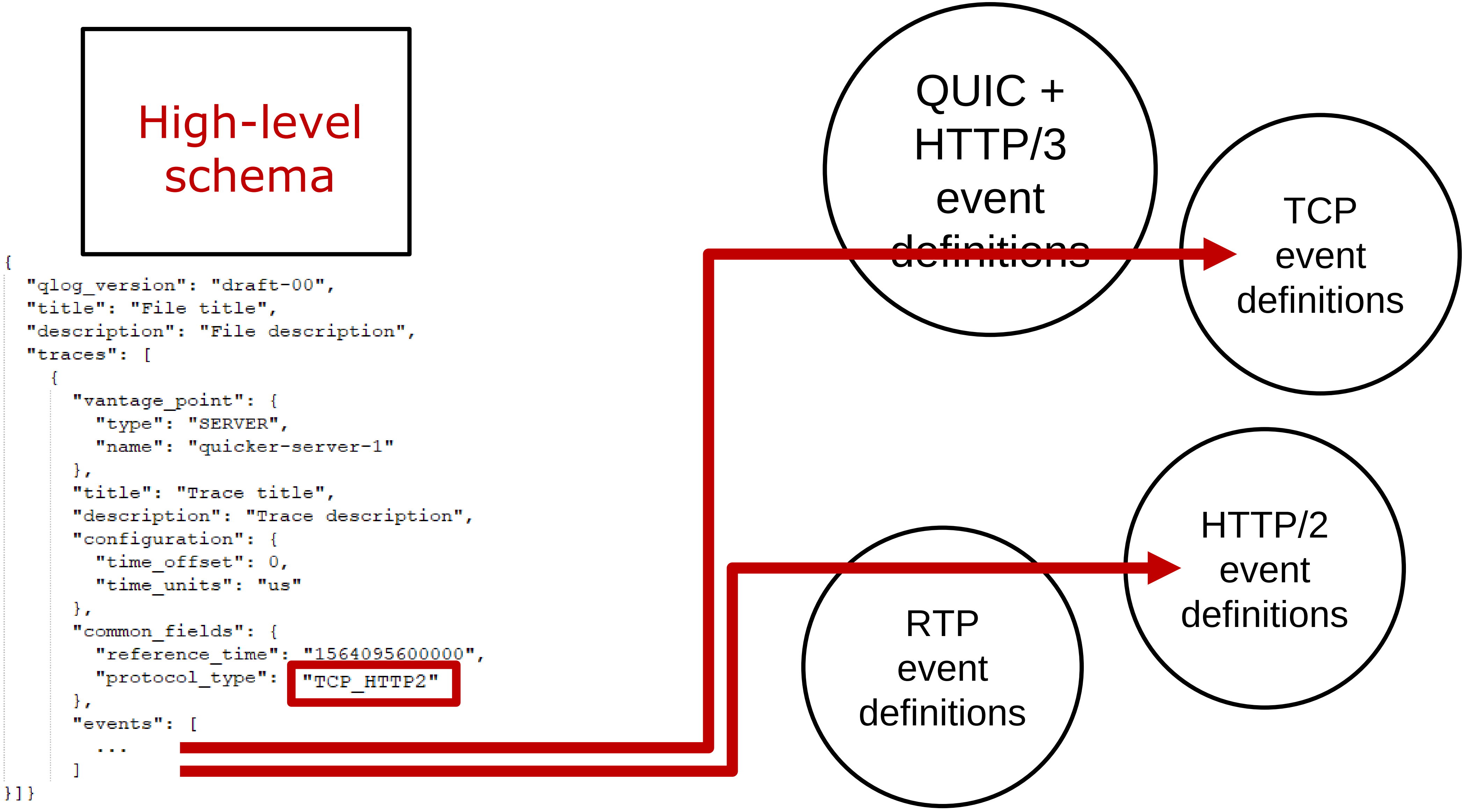


```
[
  "157487",
  "TRANSPORT",
  "PACKET_SENT",
  "DEFAULT",
  {
    "packet_type": "1RTT",
    "header": {
      "packet_number": "16",
      "packet_size": 1350
    },
    "frames": [
      {
        "frame_type": "STREAM",
        "id": "3",
        "fin": false,
        "length": 1324,
        "offset": 19850
      }
]}]
```

▶▶ UHASSELT EDM

https://tools.ietf.org/html/draft-marx-qlog-main-schema
https://tools.ietf.org/html/draft-marx-qlog-event-definitions-quic-h3

Standardized general purpose endpoint logging format, in 2 parts!



High-level schema

```
{
  "qlog_version": "draft-00",
  "title": "File title",
  "description": "File description",
  "traces": [
    {
      "vantage_point": {
        "type": "SERVER",
        "name": "quicker-server-1"
      },
      "title": "Trace title",
      "description": "Trace description",
      "configuration": {
        "time_offset": 0,
        "time_units": "us"
      },
      "common_fields": {
        "reference_time": "1564095600000",
        "protocol_type": "TCP_HTTP2"
      },
      "events": [
        ...
      ]
}]}
```

QUIC + HTTP/3 event definitions

TCP event definitions

RTP event definitions

HTTP/2 event definitions

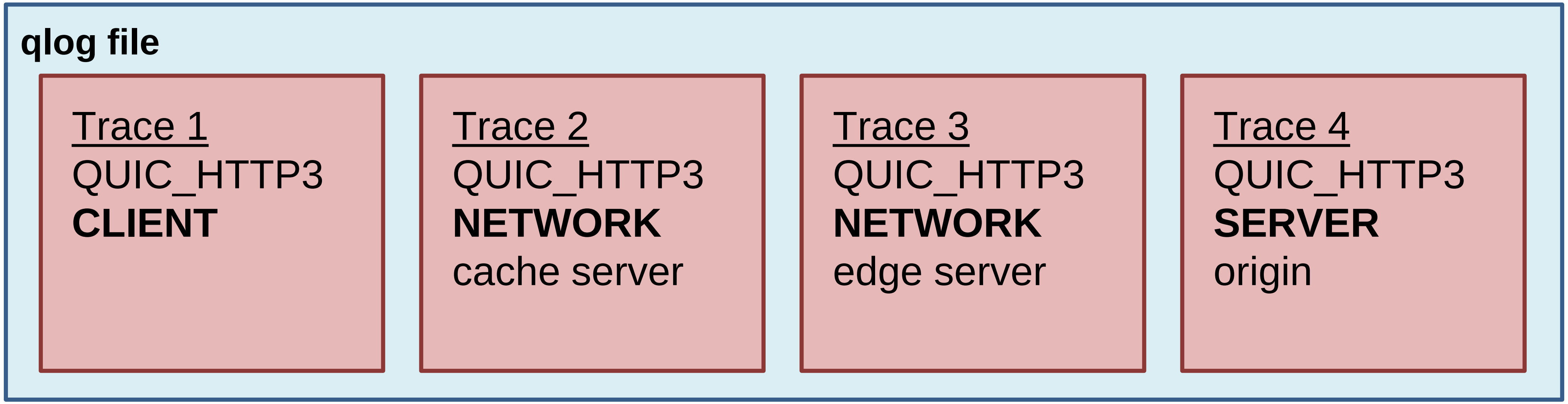
https://tools.ietf.org/html/draft-marx-qlog-main-schema
https://tools.ietf.org/html/draft-marx-qlog-event-definitions-quic-h3

High-level logging schema: support many use cases

- File size optimizations

  `common_fields` Lookup tables (log indices)

- Flexible
  - Custom categories, event types, etc. (e.g., FB data-cloned)
  - 1 file per log vs various logs aggregated in one file



qlog file

Trace 1
QUIC_HTTP3
CLIENT

Trace 2
QUIC_HTTP3
NETWORK
cache server

Trace 3
QUIC_HTTP3
NETWORK
edge server

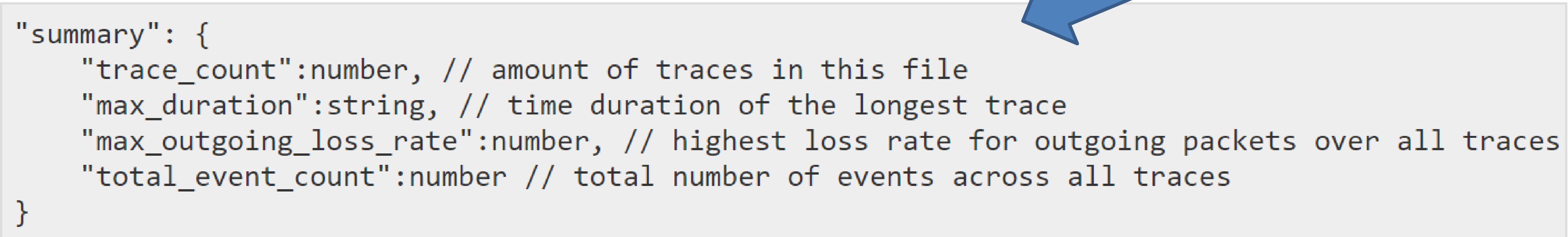
Trace 4
QUIC_HTTP3
SERVER
origin

`vantage_point`

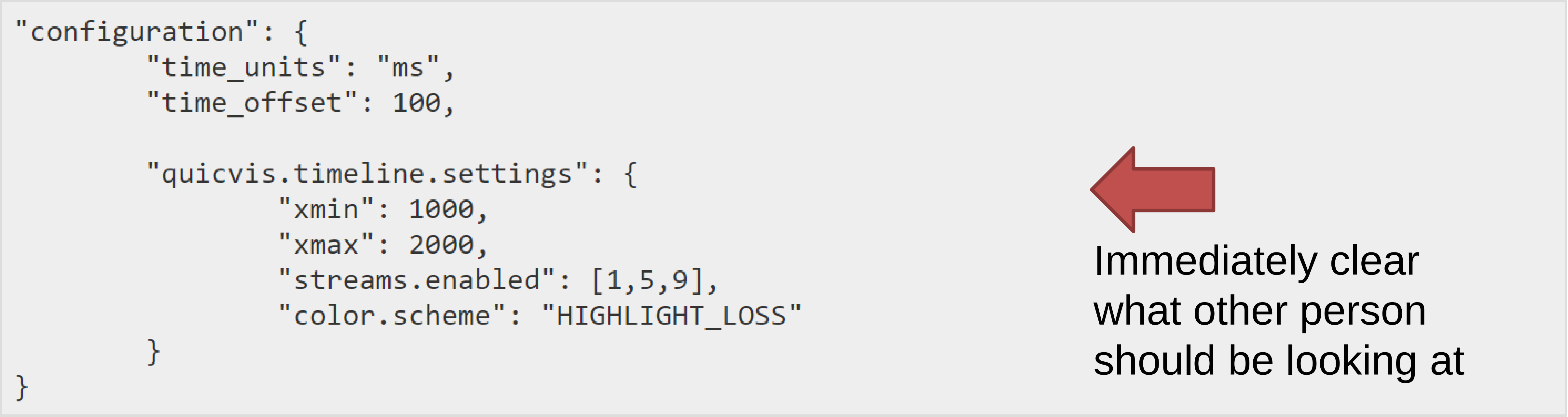
https://github.com/quiclog/internet-drafts

High-level logging schema: everything in 1 file

Quickly sift through hundreds of logs



```
"summary": {
    "trace_count":number, // amount of traces in this file
    "max_duration":string, // time duration of the longest trace
    "max_outgoing_loss_rate":number, // highest loss rate for outgoing packets over all traces
    "total_event_count":number // total number of events across all traces
}
```



```
"configuration": {
        "time_units": "ms",
        "time_offset": 100,

        "quicvis.timeline.settings": {
                "xmin": 1000,
                "xmax": 2000,
                "streams.enabled": [1,5,9],
                "color.scheme": "HIGHLIGHT_LOSS"
        }
}
```

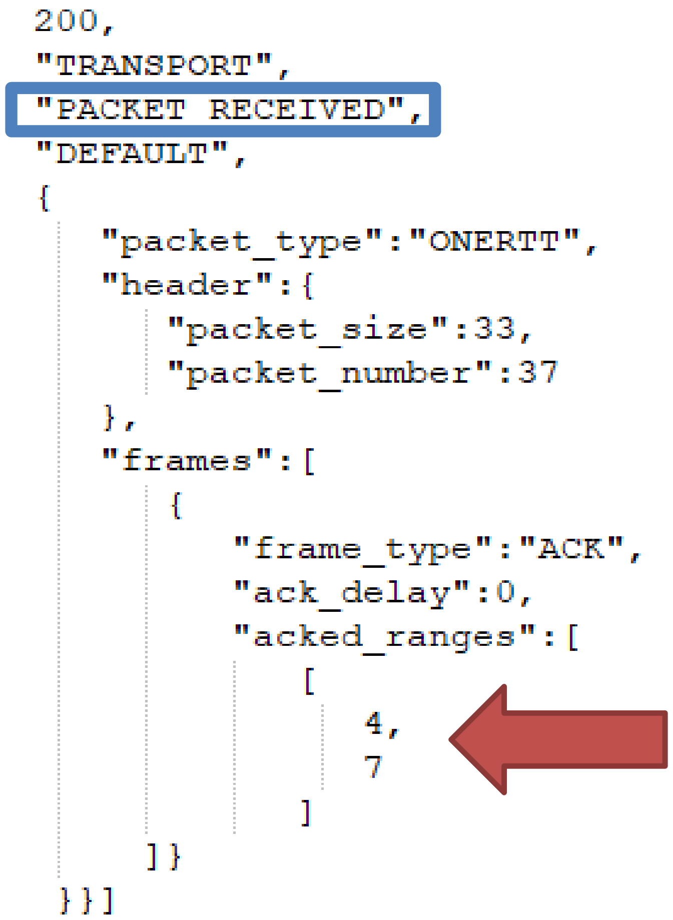
Immediately clear what other person should be looking at

QUIC and HTTP/3 event definitions: how explicit should we be?

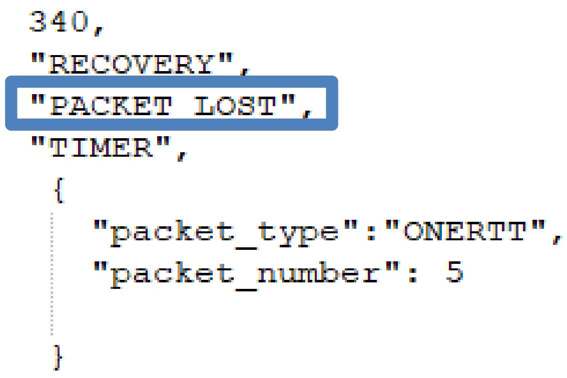
- 40+ events at the moment, will probably be 70+ by the end
    - transport.stream_state_update, recovery.packet_lost, recovery.metric_update, http.dependency_update

QUIC and HTTP/3 event definitions: how explicit should we be?

- 40+ events at the moment, will probably be 70+ by the end
  - transport.stream_state_update, recovery.packet_lost, recovery.metric_update, http.dependency_update



```
200,
"TRANSPORT",
"PACKET_RECEIVED",
"DEFAULT",
{
   "packet_type":"ONERTT",
   "header":{
      "packet_size":33,
      "packet_number":37
   },
   "frames":[
      {
         "frame_type":"ACK",
         "ack_delay":0,
         "acked_ranges":[
            [
               4,
               7
            ]
      ]}
}}]
```

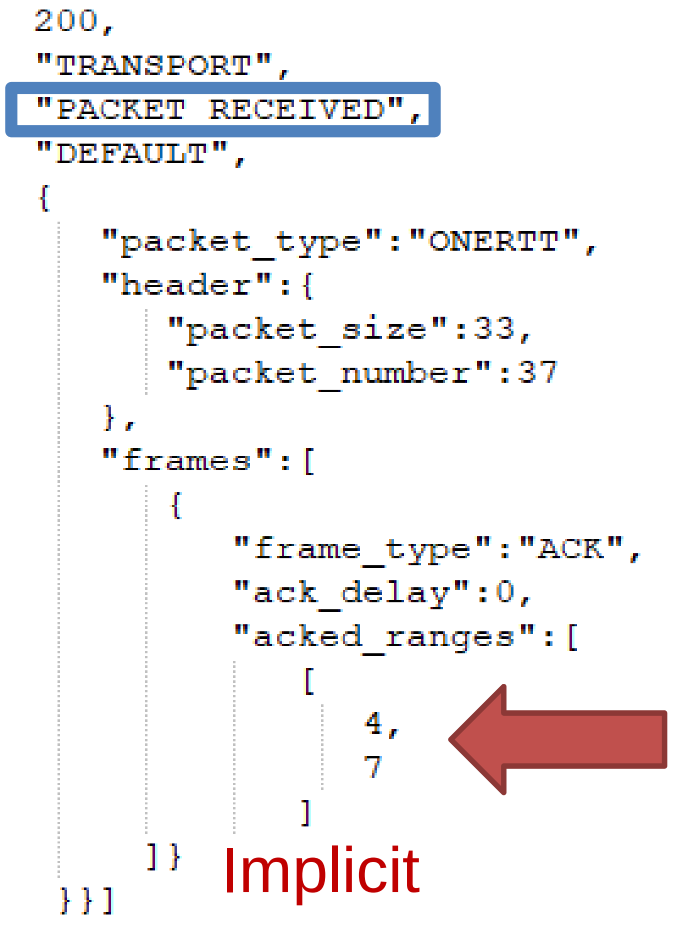


```
340,
"RECOVERY",
"PACKET_LOST",
"TIMER",
 {
   "packet_type":"ONERTT",
   "packet_number": 5

 }
```

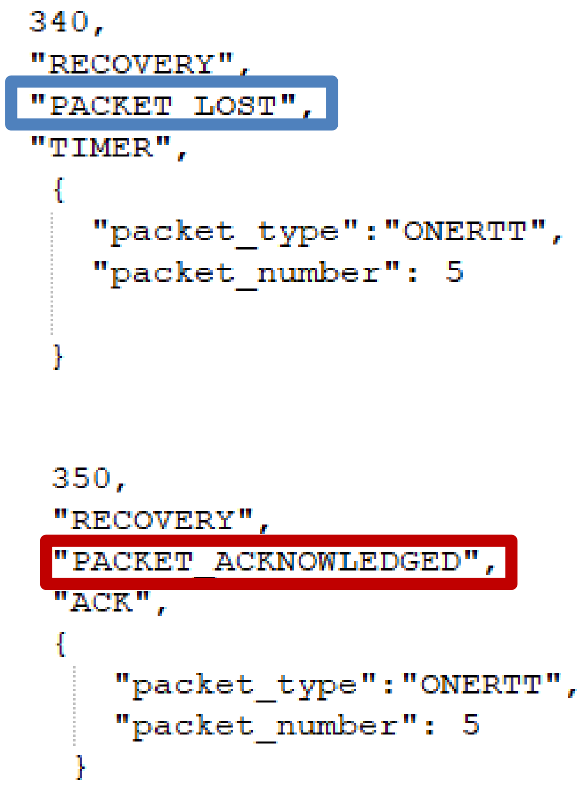
QUIC and HTTP/3 event definitions: how explicit should we be?

- 40+ events at the moment, will probably be 70+ by the end
  - transport.stream_state_update, recovery.packet_lost, recovery.metric_update, http.dependency_update



```
200,
"TRANSPORT",
"PACKET_RECEIVED",
"DEFAULT",
{
   "packet_type":"ONERTT",
   "header":{
      "packet_size":33,
      "packet_number":37
   },
   "frames":[
      {
         "frame_type":"ACK",
         "ack_delay":0,
         "acked_ranges":[
            [
               4,
               7
            ]
      ]}
}}]
```

Implicit



```
340,
"RECOVERY",
"PACKET_LOST",
"TIMER",
 {
   "packet_type":"ONERTT",
   "packet_number": 5

 }
```

```
350,
"RECOVERY",
"PACKET_ACKNOWLEDGED",
"ACK",
{
  "packet_type":"ONERTT",
  "packet_number": 5
 }
```

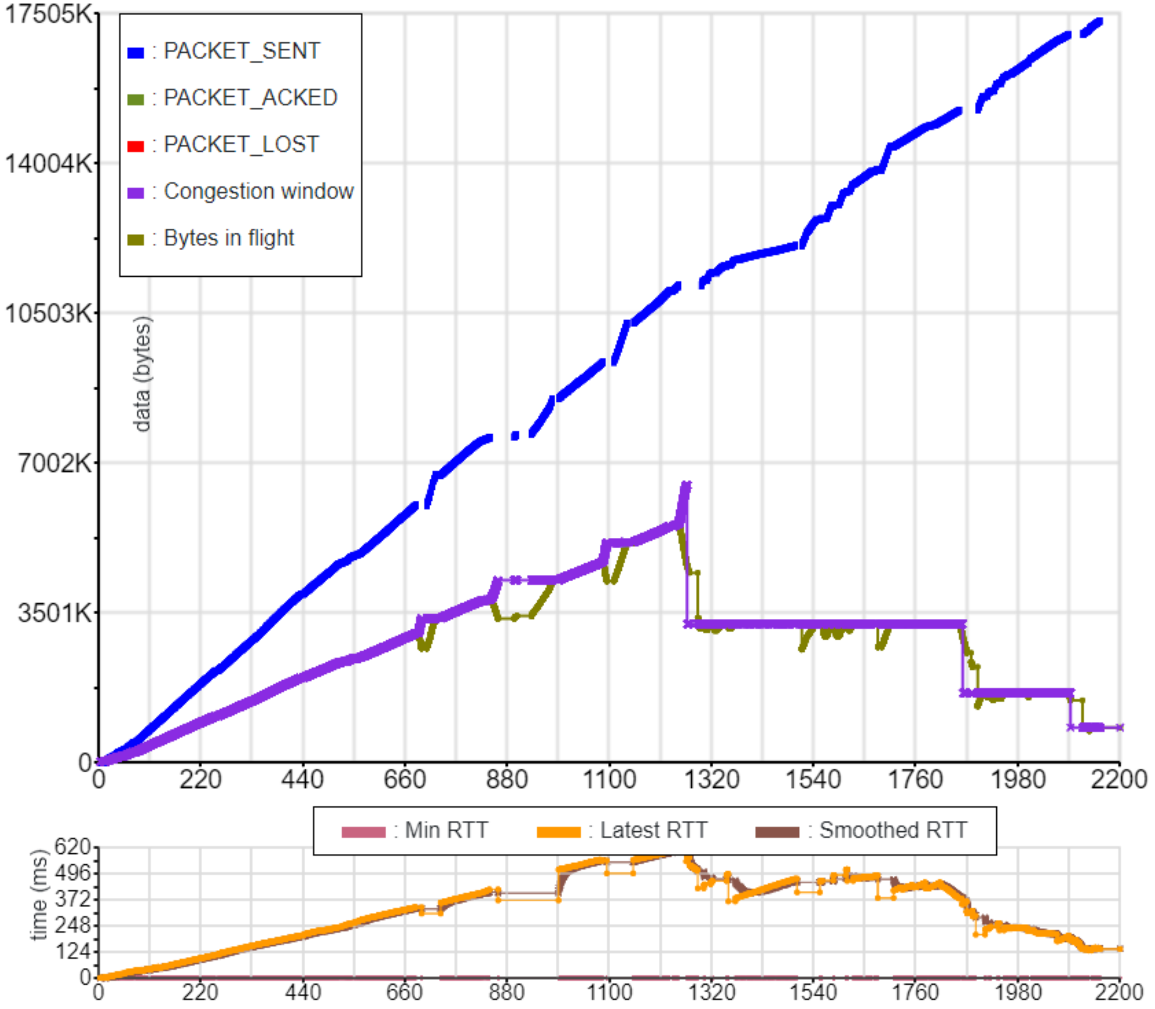
Explicit

https://www.youtube.com/watch?v=R4j7X5ktoT8&t=4729

State of the qunion

- 5 QUIC/H3 implementations output (partial) qlog directly
  - quicker
  - mvfst (facebook)
  - lsquic (litespeed)
  - quant (netapp)
  - aioquic

- 1 main public tool (congestion control), many more coming over the course of August
  - Timeline
  - Sequence diagram
  - Flow control
  - …

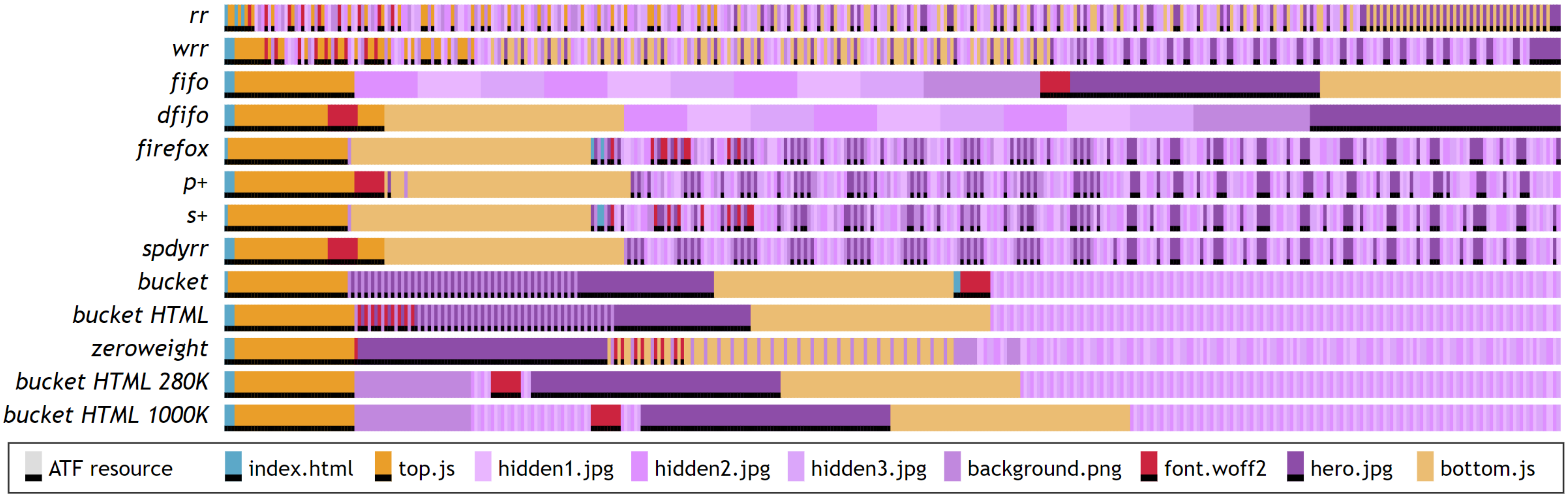


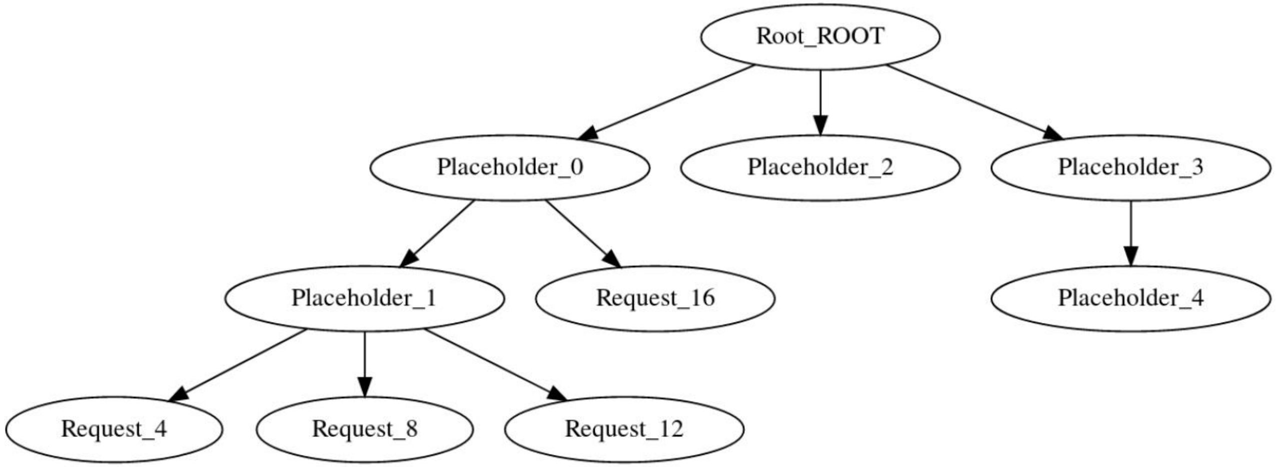
https://quic.edm.uhasselt.be/qtr-to-qlog/
https://quicvis.edm.uhasselt.be:8443

State of the qunion

- “Easy to implement, low code overhead”
- “Impact is limited: **185 Mbps** without qlog, **175** with”
- “We are logging **20 billion+** quic trace events a day” 😮
- “My test could now examine the qlog output to see whether the bit is spinning”
- “Visualizations are very useful”
- “I'm amazed by these visuals, I would never have put in the effort”

State of the qunion





State of the qunion

“Easy to implement, low code overhead”

“Impact is limited: **185 Mbps** without qlog, **175** with”

“We are logging **20 billion+** quic trace events a day” 😮

“My test could now examine the qlog output to see whether the bit is spinning”

“Visualizations are very useful”

“I'm amazed by these visuals, I would never have put in the effort”

“With the explicit events, qlog could be up to **30%** of my code”

“Enabling logging slows things down **50%+**” 😞

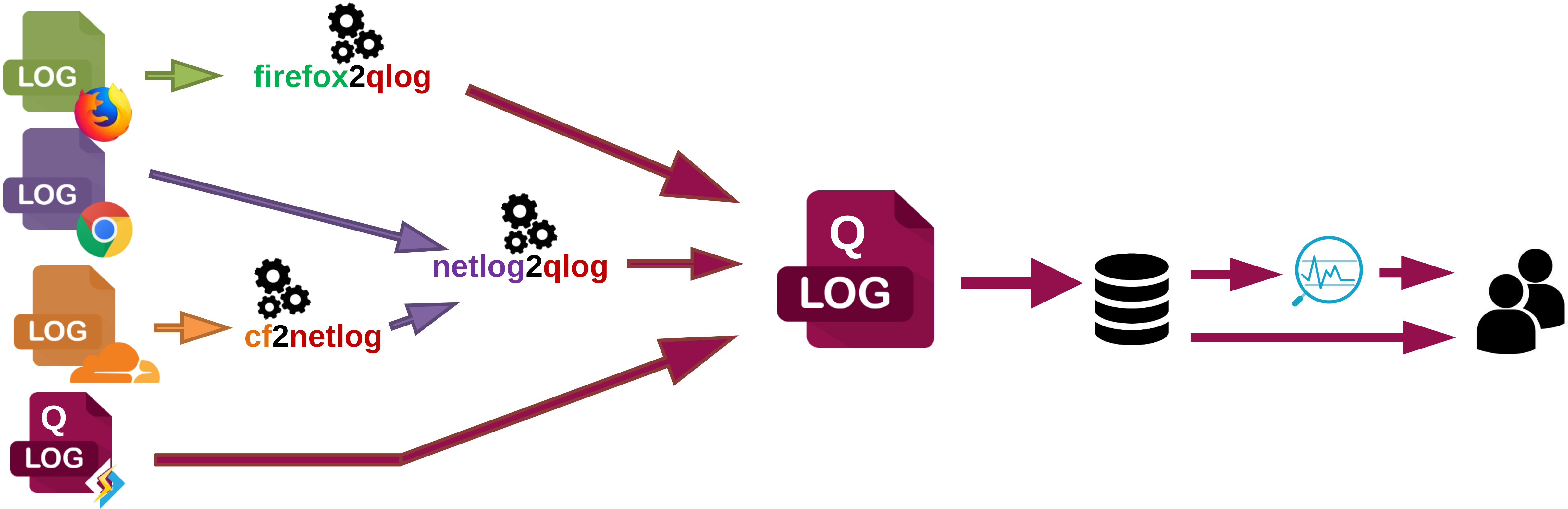
“Binary formats are much better for storage”

“I do not always have the necessary information in one place
in the code to output a specific event”

“Logging code adds maintenance and testing burden, and brings questions as
how to store/access logs”

State of the qunion

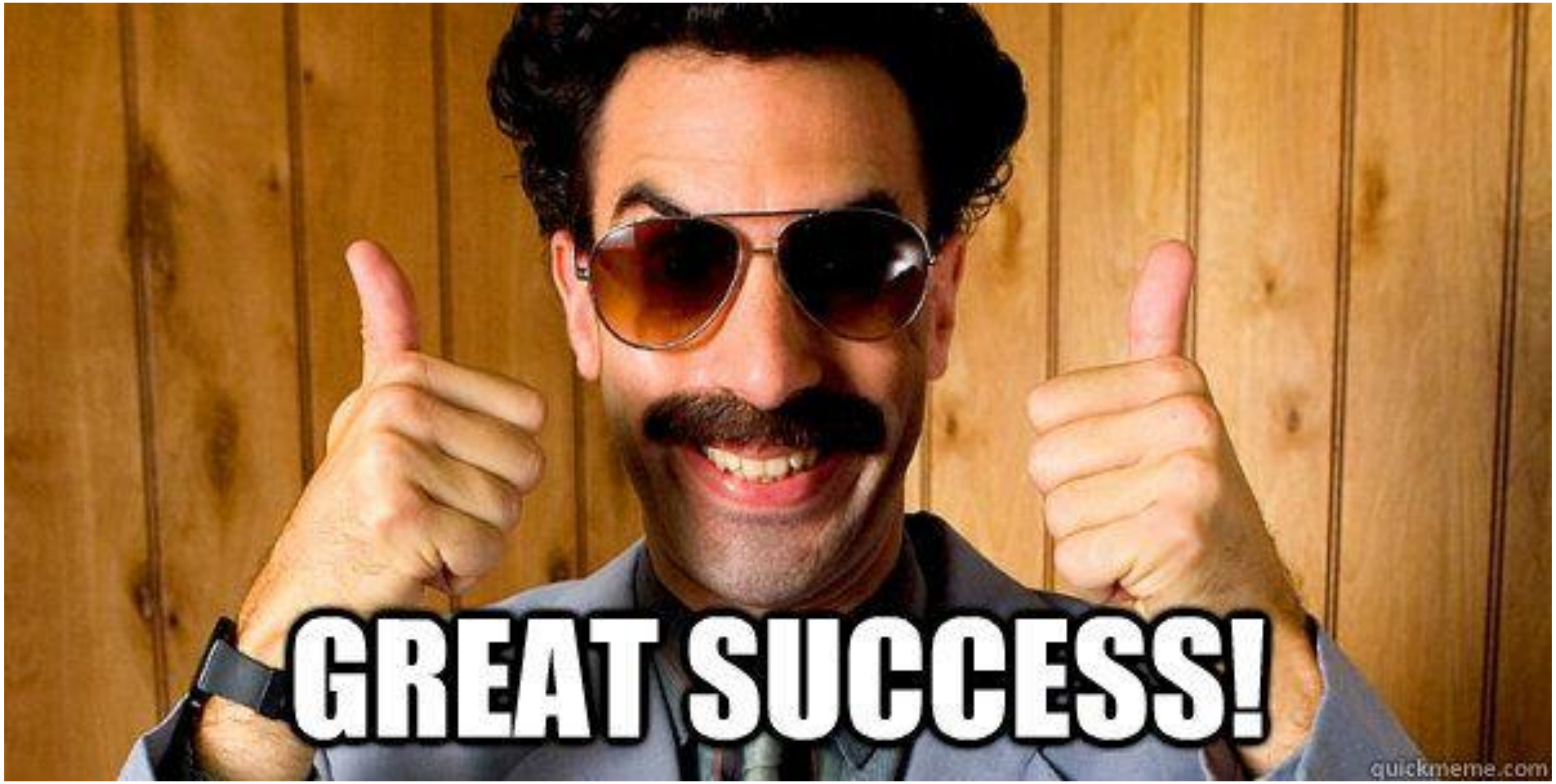
- Several transformers/converters exist
  - Pcap2qlog and quictrace2qlog
  - quiche + quicly + winquic: convert from internal format to quictrace/qlog



UHASSELT EDM

https://quic.edm.uhasselt.be/qtr-to-qlog/
https://github.com/quiclog/pcap2qlog

In conclusion



But also still many other open questions

- How flexible must the main schema be?
- Is a “trigger” field per-event useful / doable in practice?
- Preventing proliferation of someting2qlog converters?
- Fine-grainedness of events
- Privacy and security aspects

- Is this actually portable to other protocols?
- Tooling integration, log access, etc…
- What are the must-have tools?
- Does it make sense to do everything in 1 format?

Things YOU can do to help!

- Join the discussion
  - github.com/quiclog/internet-drafts
  - qlog@ietf.org
  - Soon: quictools.info

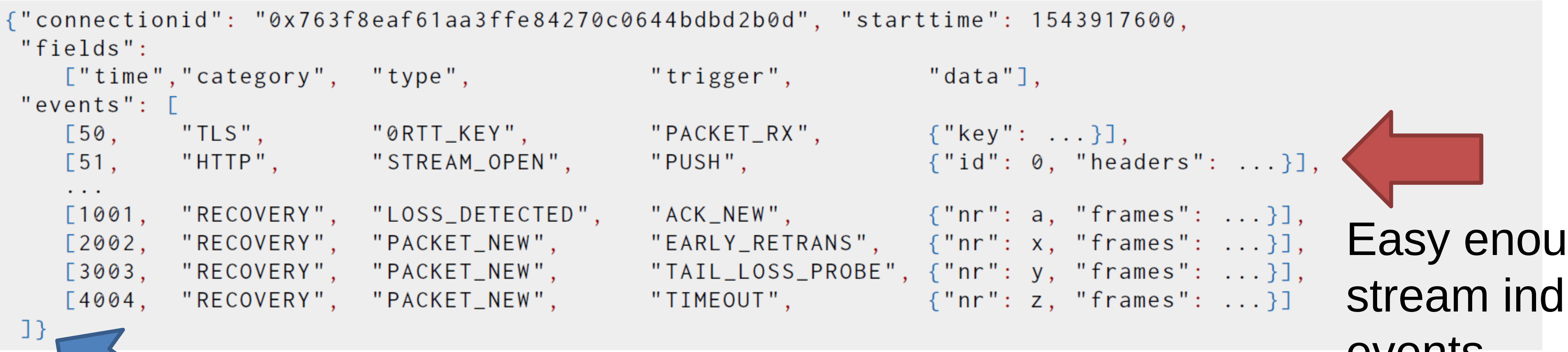
- Get your hands dirty
  - Implement qlog in your QUIC stack today!
  - Implement qlog POC for other protocols (e.g., TCP in OMNET++)
  - Implement your own visualization

- Does it make sense to move this to BOF/wg? Next steps?

Extra slides

Events can be streamed

- "Live debugging" : tool updates as events come in
- JSON is not a streamable format per se though…



```
{"connectionid": "0x763f8eaf61aa3ffe84270c0644bdbd2b0d", "starttime": 1543917600,
 "fields":
    ["time","category", "type",            "trigger",        "data"],
 "events": [
    [50,    "TLS",      "0RTT_KEY",        "PACKET_RX",      {"key": ...}],
    [51,    "HTTP",     "STREAM_OPEN",     "PUSH",           {"id": 0, "headers": ...}],
    ...
    [1001,  "RECOVERY", "LOSS_DETECTED",   "ACK_NEW",        {"nr": a, "frames": ...}],
    [2002,  "RECOVERY", "PACKET_NEW",      "EARLY_RETRANS",  {"nr": x, "frames": ...}],
    [3003,  "RECOVERY", "PACKET_NEW",      "TAIL_LOSS_PROBE", {"nr": y, "frames": ...}],
    [4004,  "RECOVERY", "PACKET_NEW",      "TIMEOUT",        {"nr": z, "frames": ...}]
 ]}
```

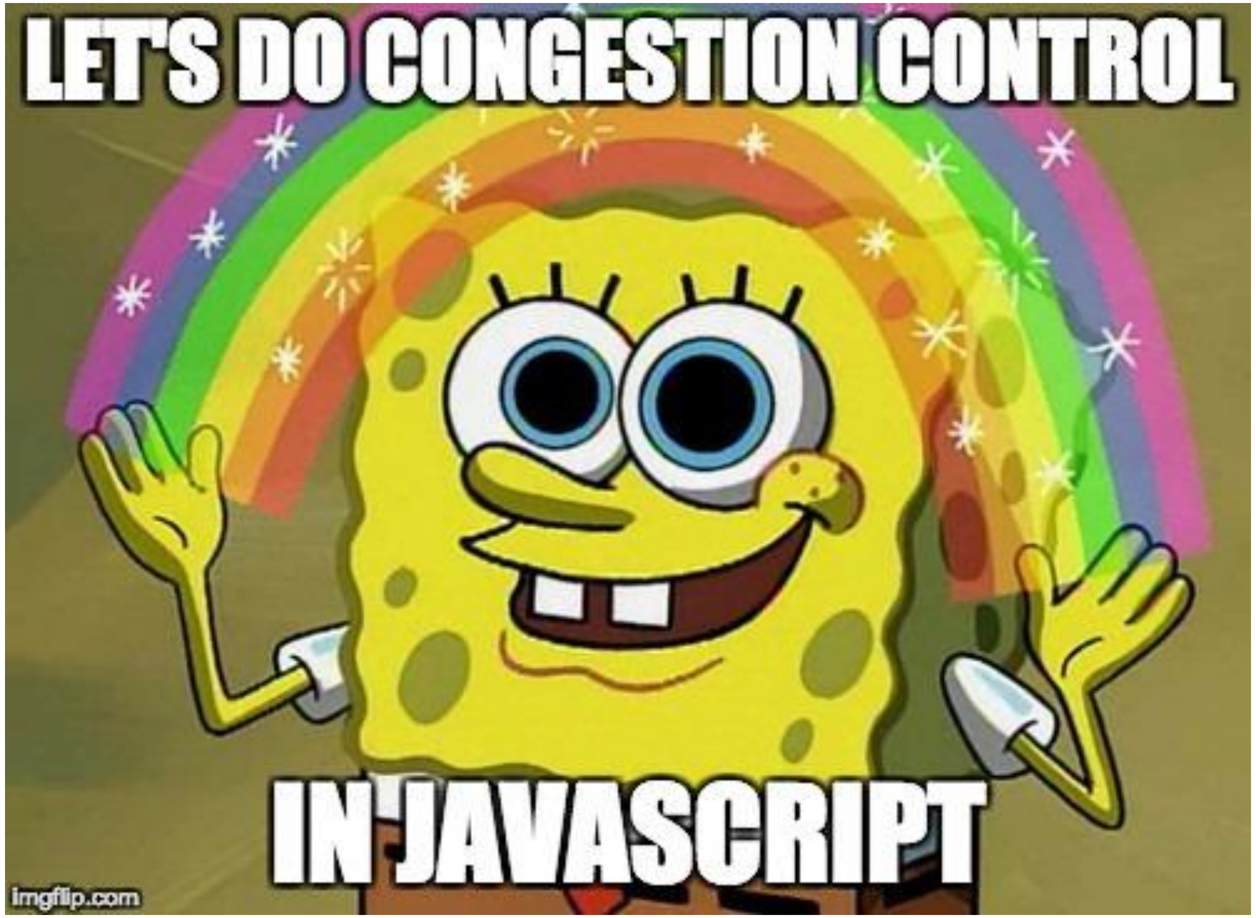
Easy enough to stream individual events

These two characters are apparently pretty important

- "Solution": streaming JSON parser

https://github.com/quiclog/internet-drafts/issues/2

More in-depth discussion in a previous talk



https://www.youtube.com/watch?v=R4j7X5ktoT8&t=4729
https://quic.edm.uhasselt.be/symposium19/