An update on qlog

Robin Marx - @programmingart
(Hasselt University – Belgium)
Motivation: QUIC and HTTP/3 tools

https://quic.edm.uhasselt.be
Motivation: QUIC logging: The Wild Wild West
Our proposal: qlog

```json
{
    "connectionid": "0x763f8eaf61aa3ffe84270c0644bd4b2b0d",
    "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", ...}],
        [203, "RECOVERY", "PACKET_QUEUE", "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

https://quic.edm.uhasselt.be
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
Standardized **general purpose** endpoint logging format, in 2 parts!

High-level schema

QUIC + HTTP/3 event definitions

```json
["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": 19050
 }
 ]
]
```

Standardized **general purpose** endpoint logging format, in 2 parts!

**High-level schema**

```json
{
  "glog_version": "draft-00",
  "title": "File title",
  "description": "File description",
  "traces": [
    {
      "vantage_point": {
        "type": "SERVER",
        "name": "quickerserver-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**

```json
["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": 19050
     }
   ]
  ]
```

Standardized general purpose endpoint logging format, in 2 parts!

High-level schema

```
{
  "glog_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": "1564895600000",
        "protocol_type": "TCP_HTTP2"
      }
    }
  ]
}
```
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

![Diagram of 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

```
"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

Quickly sift through hundreds of logs

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

https://www.youtube.com/watch?v=R4j7X5ktoT8&t=4729
State of the union

- 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 union

“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 union

https://h3.edm.uhasselt.be/
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 union

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

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

GREAT SUCCESS!
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 something2qlog 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...

```json
{"connectionid": "0x763f8eaf61aa3ffe84270c0644bd6b2d0d", "starttime": 1543917600,
  "fields": [
    {"time": "category", "type", "trigger", "data"},
  "events": [
    [50, "TLS", "ORRT_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/