structured event logging for (encrypted) protocols

Robin Marx  robin.marx@kuleuven.be
What’s in a name?

\[
\text{qlog} = \text{QUIC Logging}
\]

QUIC and HTTP/3 are complex
- Will need good debugging and analysis tools
- Tools need data to ingest

Typical network logging

get raw wire image from one location

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>TCP</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>343</td>
<td>65.142415</td>
<td>192.168.0.21</td>
<td>174.129.249.228</td>
<td>HTTP</td>
<td>253</td>
<td>84</td>
<td>GET /customers/landen/flash-application遐CurrentlyNotSupportedFlashVersion1.80v1.80m</td>
</tr>
<tr>
<td>344</td>
<td>65.142715</td>
<td>192.168.0.21</td>
<td>174.129.249.228</td>
<td>HTTP</td>
<td>253</td>
<td>84</td>
<td>GET /customers/landen/flash-application遐CurrentlyNotSupportedFlashVersion1.80v1.80m</td>
</tr>
<tr>
<td>345</td>
<td>65.220730</td>
<td>174.129.249.228</td>
<td>192.168.0.21</td>
<td>HTTP</td>
<td>253</td>
<td>84</td>
<td>GET /customers/landen/flash-application遐CurrentlyNotSupportedFlashVersion1.80v1.80m</td>
</tr>
<tr>
<td>346</td>
<td>65.249742</td>
<td>174.129.249.228</td>
<td>10.163.8.21</td>
<td>HTTP</td>
<td>253</td>
<td>84</td>
<td>GET /customers/landen/flash-application遐CurrentlyNotSupportedFlashVersion1.80v1.80m</td>
</tr>
<tr>
<td>347</td>
<td>65.245392</td>
<td>192.168.0.21</td>
<td>174.129.249.228</td>
<td>HTTP</td>
<td>253</td>
<td>84</td>
<td>GET /customers/landen/flash-application遐CurrentlyNotSupportedFlashVersion1.80v1.80m</td>
</tr>
<tr>
<td>348</td>
<td>65.275376</td>
<td>192.168.0.21</td>
<td>120.168.0.21</td>
<td>DNS</td>
<td>77</td>
<td>22</td>
<td>Standard query response 0x180 A cdn-0.mflling.com (RUC) images.netflix.com.edge</td>
</tr>
<tr>
<td>350</td>
<td>65.377953</td>
<td>63.52.52.48</td>
<td>63.98.242.48</td>
<td>TCP</td>
<td>74</td>
<td>37683</td>
<td>300 [SYN, ACK] Seq=9101936 Ack=158588 Len=0 Tsal=4915534240 TSecr=5513118827</td>
</tr>
<tr>
<td>351</td>
<td>65.297757</td>
<td>63.80.242.48</td>
<td>63.98.242.48</td>
<td>TCP</td>
<td>74</td>
<td>37683</td>
<td>300 [SYN, ACK] Seq=9101936 Ack=158588 Len=0 Tsal=4915534240 TSecr=5513118827</td>
</tr>
<tr>
<td>352</td>
<td>65.298966</td>
<td>192.168.0.21</td>
<td>10.163.8.21</td>
<td>HTTP</td>
<td>253</td>
<td>84</td>
<td>GET /documents/flash/flash-content.v1.0 HTTP/1.1</td>
</tr>
<tr>
<td>353</td>
<td>65.296887</td>
<td>10.163.8.21</td>
<td>174.129.249.228</td>
<td>HTTP</td>
<td>253</td>
<td>84</td>
<td>GET /documents/flash/flash-content.v1.0 HTTP/1.1</td>
</tr>
<tr>
<td>354</td>
<td>65.318730</td>
<td>63.80.242.48</td>
<td>63.98.242.48</td>
<td>TCP</td>
<td>74</td>
<td>37683</td>
<td>300 [SYN, ACK] Seq=9101936 Ack=158588 Len=0 Tsal=4915534240 TSecr=5513118827</td>
</tr>
<tr>
<td>355</td>
<td>65.323735</td>
<td>63.80.242.48</td>
<td>63.98.242.48</td>
<td>TCP</td>
<td>74</td>
<td>37683</td>
<td>300 [SYN, ACK] Seq=9101936 Ack=158588 Len=0 Tsal=4915534240 TSecr=5513118827</td>
</tr>
</tbody>
</table>
1. QUIC is almost entirely encrypted

Storing full packet captures and TLS secrets is bad for:
- scalability
- privacy

<table>
<thead>
<tr>
<th>TCP</th>
<th>Encrypted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Src Port</td>
<td>Dest Port</td>
</tr>
<tr>
<td>Seq No</td>
<td>ACK No</td>
</tr>
<tr>
<td>Flags</td>
<td>Windows</td>
</tr>
<tr>
<td>Options</td>
<td>Payload</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UDP</th>
<th>QUIC (open)</th>
<th>QUIC (encrypted)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Src Port</td>
<td>Dest Port</td>
<td></td>
</tr>
<tr>
<td>Flags</td>
<td>Connection ID</td>
<td>Packet No</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Frame</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ACK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Window</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Options</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Payload</td>
</tr>
</tbody>
</table>
1. QUIC is almost entirely encrypted

2. not everything is sent on the wire
   congestion control, decision making, internal errors, ...

img src: https://labs.apnic.net/?p=1207
structured endpoint logging

get data from (both) implementations directly
Event examples

```
{
  "time": 15000,
  "name": "transport:packet_received",
  "data": {
    "header": {
      "packet_type": "1rtt",
      "packet_number": 25
    },
    "frames": [
      {
        "frame_type": "ack",
        "acked_ranges": [
          [10,15],
          [17,20]
        ]
      }
    ]
  }
}
```

```
{
  "time": 15001,
  "name": "recovery:metrics_updated",
  "data": {
    "min_rtt": 25,
    "smoothed_rtt": 30,
    "latest_rtt": 25,
    "congestion_window": 60,
    "bytes_in_flight": 77000,
  }
}
```
QUIC and HTTP/3 tools

https://github.com/quiclog/qvis

https://qvis.quictools.info
“TCPtrace” for QUIC

https://qvis.quictools.info

https://github.com/quiclog/qvis
> 75% of QUIC/H3 stacks support direct qlog output:

- mvfst
- ngtcp2
- quiche
- quic-go
- aioquic
- quicly / H2O
- neqo
- picoquic

Facebook
Node.js
Cloudflare
Fastly
Mozilla

mjoras 10:35 PM
@rmarsx we currently have qlog enabled in prod with similar amounts of events being recorded a day as I quoted before (dozens of billions).

https://qlog.edm.uhasselt.be/anrw
qlog draft adoption in QUIC wg
- Expected before or during IETF 111
- Part of recharter

Goals
- Flesh out schema’s for QUIC and HTTP/3

- Prepare qlog for broader use with other protocols / applications
  - TCP + TLS + HTTP/x
  - DNS, BGP, WebTransport
  - Multipath TCP and QUIC, MASQUE
  - Adaptive BitRate (ABR) video streaming logic
  - ...

Main
- Protocol-agnostic
  - Container / metadata
  - Format (JSON)
  - Best practices / guidelines

QUIC
- Connectivity
- Transport
- Recovery

HTTP/3
- HTTP/3
- QPACK

... Hopefully more to come

Plenty of challenges

- Event definitions
- Formats and datatypes
- Privacy and security aspects

- Operational aspects
- Cross-protocol tooling
- Protocol overlaps (e.g., TCP and QUIC, HTTP/3 vs HTTP/2 and 1, DoX, ...)
- ...
Event definitions

```json
{
  "time": 15000,
  "name": "transport:packet_received",
  "data": {
    "header": {
      "packet_type": "1rtt",
      "packet_number": 25
    },
    "frames": [
      {
        "frame_type": "ack",
        "ackedRanges": [
          [10, 15],
          [17, 20]
        ]
      }
    ]
  }
}
```
Event definitions

```json
{
  "time": 15000,
  "name": "transport:packet_received",
  "data": {
    "header": {
      "packet_type": "1rtt",
      "packet_number": 25
    },
    "frames": [
      {
        "frame_type": "ack",
        "acked_ranges": [
          [10, 15],
          [17, 20]
        ]
      }
    ]
  }
}
```

and / or?

```
{
  "time": 15000,
  "name": "transport:packets_acked",
  "data": {
    "packets": [
      19,
      20
    ]
  }
}
```

```
{
  "time": 15000,
  "name": "transport:packets_lost",
  "data": {
    "packets": [
      16
    ]
  }
}
```
Event definitions

TCP wire image

```json
{
    "time": 15000,
    "name": "transport:packet_received",
    "data": {
        "header": {
            "seq_number": 25,
            "options": [
                {
                    "type": "sack",
                    "acked_ranges": [
                        [10,15],
                        [17,20]
                    ]
                }
            ]
        }
    }
}
```

and / or?

```json
{
    "time": 15000,
    "name": "transport:packets_acked",
    "data": {
        "packets": [19,20]
    }
}
```

```json
{
    "time": 15000,
    "name": "transport:packets_lost",
    "data": {
        "packets": [16]
    }
}
```
**qlog** serialization format

**qlog is currently JSON-based**
- 500 MB transfer → 300 MB qlog
- With compression: 18 MB

**Format agnostic**
- Define datatypes and schema
- Can be mapped to multiple serialization formats
  - Which one(s) should we focus on?
  - Automated generation from text?

**Stream vs file-based**
- Typical ingestion/storage/analysis pipelines

```javascript
class StreamFrame{
  frame_type:string = "stream";
  stream_id:uint64;
  offset:uint64;
  length:uint64;
  fin?:boolean;
  raw?:bytes;
}
```

https://github.com/quiclog/internet-drafts/issues/30
Lots of sensitive data
- IP addresses / Connection IDs
- HTTP payloads, SNIs
- Timestamps?

“Sanitization levels”
- From loose to strict
- Concrete guidelines and rules
- Tagging of individual fields

Log Storage/Transport/Sharing
- Encrypt logs themselves?
- Safe access to external log sources (e.g., QUIC manageability, research datasets)
Next steps

Eventually:
- Separate qlog wg for main aspects?
- Individual (protocol) wg’s define new qlog documents?

First step:
- Drafts adoption in the QUIC wg (part of recharter)
- Expected before or during IETF 111

In the mean time
- Join us on github.com/quiclog/internet-drafts
- Join the qlog IETF mailing list ietf.org/mailman/listinfo/qlog

Give feedback now!