DNS-over-QUIC
(DoQ)
draft-ietf-dprive-dnsoquic

Christian Huitema
Sara Dickinson
Allison Mankin
DNS-over-QUIC

● Why are we standardizing ANOTHER protocol for encrypted DNS?

● Where are we with the IETF standards process?

● Where are we with implementation and deployment?
  ○ Next steps?
QUIC - Background

- QUIC and HTTP/3 developed by Google as experiment in 2012
- Development moved to IETF in 2015, standardized in 2021
- Deployed by browsers and CDNs (7.6% websites)
QUIC - Background

- QUIC and HTTP/3 developed by Google as experiment in 2012
- Development moved to IETF in 2015, standardized in 2021
- Deployed by browsers and CDNs (7.6% websites)

**Key QUIC characteristics**
- TLS 1.3 secured transport that runs over UDP
- Reduced latency in handshake (0-RTT)
- Stream based - no head of line blocking
- Improved error detection and loss recovery compared to TCP
- Connection migration (IP address can change)

- HTTP/3 runs over QUIC
DoQ - Background

- Early realisation that DoQ would be a good fit for encrypted DNS
  - Low latency
  - UDP but with QUIC benefits and
    - Source address validation
    - Path MTU does not limit size of messages
- But… QUIC WG decided QUIC v1 would only support HTTP/3
DoQ - Background

- **April 2017** - First Draft in QUIC WG
- **December 2018** - Adguard DoQ service launched

- **Apr 2020** - Draft adopted in DPRIVE WG (stub to rec ONLY)
- **2021** - Re-scoped to include XFR and rec to auth, mapping updated and port 853 requested (more later)
- **Oct-Dec 2021** - Working group last call
- **January 2022** - IETF Last Call (RFC later this year?)
What does DoQ look like?

- Set up a connection with a QUIC handshake (TLS 1.3)
- Use ALPN ‘doq’

Images from https://blog.cloudflare.com/the-road-to-quic/
What does DoQ look like?

- Exchange of messages on streams (ids are 4, 8, 12)
- One stream is used for one DNS query/response (then closed)
- There are $2^{64}$ stream IDs - that’s a lot of messages on one connection
  - MessageID is ALWAYS 0
What does DoQ look like?

- Exchange of messages on streams (ids are 4, 8, 12)
- One stream is used for one DNS query/response (then closed)
- There are $2^{64}$ stream ID - that’s a lot of messages on one connection
  - MessageID is ALWAYS 0
- Original mapping
  - JUST DNS message
  - Immediate close at both ends

![Single QUIC connection diagram]

<table>
<thead>
<tr>
<th>STREAM 4: Query</th>
<th>STREAM 8: Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>STREAM 8: Response</td>
<td></td>
</tr>
<tr>
<td>STREAM 4: Response</td>
<td></td>
</tr>
</tbody>
</table>
But.. if we want to do XFR…

- **Original mapping**

- **Current mapping**
  - Prepend with length field (like TCP)
  - Server can send multiple responses
DoQ is a general purpose protocol

- AdGuard claim **good performance** (particularly in mobile networks)
- With the more flexible mapping **XFR is now possible** (RFC for XFR-over-TLS last year)
- **Lots of interest in using DoQ for recursive to authoritative**
  - Lighter weight than DoT and DoH, better latency
  - Protocol is maturing
DoQ is a general purpose protocol

- AdGuard claim **good performance** (particularly in mobile networks)
- With the more flexible mapping **XFR is now possible** (RFC for XFR-over-TLS last year)
- **Lots of interest in using DoQ for recursive to authoritative**
  - Lighter weight than DoT and DoH, better latency
  - Protocol is maturing

- Originally port 784 was used for experiments but WG proposed to use port 853 (assigned to DNS-over-DTLS in 2016). IANA process but…. now agreed.
  - TCP port 853: DNS-over-TLS
  - **UDP port 853: DNS over DTLS or QUIC**
# DoQ Implementations (open source)

<table>
<thead>
<tr>
<th>Implementation</th>
<th>Language</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CoreDNS</td>
<td>Go</td>
<td>AdGuard use as DoQ server</td>
</tr>
<tr>
<td>AdGuard DNS Proxy</td>
<td>Go</td>
<td>Simple proxy or server supporting DoQ (used in ADGuard Home)</td>
</tr>
<tr>
<td>dnslookup</td>
<td>Go</td>
<td>Command line utility wrapper for Adguard DNS proxy</td>
</tr>
<tr>
<td>AdGuard C++ DNS libs</td>
<td>C++</td>
<td>AdGuard use in mobile app</td>
</tr>
<tr>
<td>Quicdoc</td>
<td>C</td>
<td>Simple DoQ impl based on Picoquic</td>
</tr>
<tr>
<td>aioquic</td>
<td>Python</td>
<td>QUIC implementation includes example DoQ client/server</td>
</tr>
<tr>
<td>Flamethrower</td>
<td>C++</td>
<td>DNS performance utility with experimental DoQ</td>
</tr>
</tbody>
</table>

- Starting to hear interest in recursive to auth experiments
DoQ open issues

- **Padding**
  - Multiple studies show padding is needed or ML can derive the encrypted queries
  - Discussion: Require implementation of the Experimental RFC 8467
  - This will be a new IETF Last Call to approve a Down Reference for RFC 8467 to be Normative

- **Security Considerations**
  - Does community see missing considerations for recursive to auth?
    - Note that authentication model for rec to auth are still a WIP
    - Privacy vs Security is a tricky trade-off for DoQ

- Lacking formal performance measurements (particularly for rec to auth traffic patterns)
DoQ open issues

- **Padding**
  - Multiple studies show padding is needed or ML can derive the encrypted queries
  - Discussion: Require implementation of the Experimental RFC 8467
  - This will be a new IETF Last Call to approve a Down Reference for RFC 8467 to be Normative

- **Security Considerations**
  - Does community see missing considerations for recursive to auth?
    - Note that authentication model for rec to auth are still a WIP
    - Privacy vs Security is a tricky trade-off for DoQ

- Lacking formal performance measurements (particularly for rec to auth traffic patterns)

Please review the specification!
Backup slides
DoQ vs DoT vs DoH3?

- **DoQ**: QUIC over UDP
- **DoT**: TLS over TCP
- **DoH3**: HTTP-3 over UDP