DNS Queries over CoAP (DoC)

draft-lenders-dns-over-coap
(https://datatracker.ietf.org/doc/draft-lenders-dns-over-coap/)

Martine S. Lenders, Christian Amsüss, Cenk Gündoğan,
Thomas C. Schmidt, Matthias Wählisch
IETF 113 CoRE Meeting, 2022-03-25
Outline

Introduction

Update since interim-2021-core-12

Preliminary evaluation

Discussion

A new Content-Format

Caching and Max-Age vs. DNS TTL

Do we need to account for OBSERVE/Server Push?

How abstract should the draft be?
Motivation

Attack Scenario

Countermeasure: Encrypt name resolution triggered by IoT devices
Possible solutions:

- DNS over HTTPS (RFC 8484)
- DNS over TLS (RFC 7858)
- DNS over QUIC (draft)
- DNS over DTLS (RFC 8094)

TCP conflicts with resource constraints
TLS over UDP conflicts with resource constraints
Path MTU problem vs constrained link layer PDUs

Our proposal: DNS over CoAP
- Encrypted communication based on DTLS or OSCORE
- Block-wise message transfer to overcome Path MTU problem
- Share system resources with CoAP applications
- Same socket and buffers can be used
- Re-use of the CoAP retransmission mechanism
Possible solutions:

- DNS over HTTPS (RFC 8484)
- DNS over TLS (RFC 7858)
- DNS over QUIC (draft)

Our proposal: DNS over CoAP

- Encrypted communication based on DTLS or OSCORE
- Block-wise message transfer to overcome Path MTU problem
- Share system resources with CoAP applications
- Same socket and buffers can be used
- Re-use of the CoAP retransmission mechanism
Possible solutions:

- DNS over HTTPS (RFC 8484)
- DNS over TLS (RFC 7858)
- DNS over QUIC (dprive draft)
- DNS over DTLS (RFC 8094)

TCP conflicts with resource constraints
TLS over UDP conflicts with resource constraints
Path MTU problem vs constrained link layer PDUs

Our proposal: DNS over CoAP
- Encrypted communication based on DTLS or OSCORE
- Block-wise message transfer to overcome Path MTU problem
- Share system resources with CoAP applications
- Same socket and buffers can be used
- Re-use of the CoAP retransmission mechanism
Possible solutions:

- DNS over HTTPS (RFC 8484)
- DNS over TLS (RFC 7858)
- DNS over QUIC (dprive draft)
- DNS over DTLS (RFC 8094)

TCP conflicts with resource constraints
Possible solutions:

DNS over HTTPS (RFC 8484)

DNS over TLS (RFC 7858)

DNS over QUIC (draft)

DNS over DTLS (RFC 8094)

TCP conflicts with resource constraints

TLS over UDP conflicts with resource constraints

Our proposal: DNS over CoAP
- Encrypted communication based on DTLS or OSCORE
- Block-wise message transfer to overcome Path MTU problem
- Share system resources with CoAP applications
- Same socket and buffers can be used
- Re-use of the CoAP retransmission mechanism
Possible solutions:

DNS over HTTPS (RFC 8484)

DNS over TLS (RFC 7858)

DNS over QUIC (draft)

DNS over DTLS (RFC 8094)

TCP conflicts with resource constraints

TLS over UDP conflicts with resource constraints

Path MTU problem vs constrained link layer PDUs

Our proposal: DNS over CoAP

• Encrypted communication based on DTLS or OSCORE

• Block-wise message transfer to overcome Path MTU problem

• Share system resources with CoAP applications

• Same socket and buffers can be used

• Re-use of the CoAP retransmission mechanism
Possible solutions:

Possible solutions:

- DNS over HTTPS (RFC 8484)
- DNS over TLS (RFC 7858)
- DNS over QUIC (draft)
- DNS over DTLS (RFC 8094)

TCP conflicts with resource constraints
TLS over UDP conflicts with resource constraints
Path MTU problem vs constrained link layer PDUs

Our proposal: DNS over CoAP

- Encrypted communication based on DTLS or OSCORE
- Block-wise message transfer to overcome Path MTU problem
- Share system resources with CoAP applications
  - Same socket and buffers can be used
  - Re-use of the CoAP retransmission mechanism
Overview

- FETCH coaps://[2001:db8::1]/

CoAP request

+--------+ [DNS query] +--------+ DNS query +--------+
| DoC |----------------| DoC |...............| DNS |
| Client |<----------------| Server |<...............| Server |

+--------+ CoAP response +--------+ DNS response +--------+
[DNS response]
What happened since interim-2021-core-12

draft-lenders-dns-over-coap-02

- Remove GET and POST method specification
- Add note on ETag and response codes
- Clarify why DoQ conflicts with constrained IoT scenarios
- Clarify Content-Format / Accept handling

draft-lenders-dns-over-coap-03

- Clarify server selection to be out-of-band
- Define ”core.dns” resource type
- Add considerations on message manipulation for DoC servers
- Update considerations on unencrypted use
Evaluation: Setup

**Name properties:** Based on empirically measured data from IoT devices

**Testbed experiments:**

- Clients query 50 A or AAAA records for names of length 24 chars via DNS over UDP / DTLSv1.2 / CoAP (unencrypted) / CoAPSv1.2 / OSCORE
- Poisson distribution: $\lambda = 5$ queries / sec (ignoring NSTART=1 requirements)
- 10 runs on IoT-nodes (incl. BR): Cortex-M3 with IEEE 802.15.4 radio
Experiment: Resolution time

DNS Transports
- UDP
- DTLSv1.2
- CoAP
- CoAPSv1.2
- OSCORE

CoAP Methods
- POST
- GET
- FETCH

CDF
A record
Resolution time [s]
0.0
0.5
1.0

AAAA record
Resolution time [s]
0 10 20 30 40

Packet size
L2 max. frame size
0 32 64 96 128 160 192
Frame Size [bytes]
UDP
0 32 64 96 128 160 192

DTLSv1.2
0 32 64 96 128 160 192

CoAP
0 32 64 96 128 160 192

CoAPSv1.2

OSCORE
Packet size L2 max. frame size
0 32 64 96 128 160 192
Frame Size [bytes]
Experiment: Resolution time

Clear performance groupings visible
Experiment: Resolution time

 DNS Transports
- UDP

 A record

 AAAA record

 Group 1
Experiment: Resolution time & packet sizes

DNS Transports
- UDP
- CoAP

CoAP Methods
- POST
- FETCH

A record
AAAA record

CDF

Resolution time [s]

Frame Size [bytes]

UDP
DTLSv1.2
CoAP
CoAPSv1.2
OSCORE

CoAP Methods
- POST
- FETCH

DNS Transports
- UDP
- DTLSv1.2
- CoAP
- CoAPSv1.2
- OSCORE

Group 2
Experiment: Resolution time & packet sizes

A record

- DNS Transports
  - CoAPSv1.2
  - DTLSv1.2
  - OSCORE
  - CoAP

- CoAP Methods
  - POST
  - GET
  - FETCH

AAAA record

- DNS Transports
  - UDP
  - DTLSv1.2
  - CoAP
  - CoAPSv1.2
  - OSCORE

- CoAP Methods
  - POST
  - GET
  - FETCH

Group 3
Experiment: Resolution time & packet sizes

Where do performance groups come from?
Experiment: Resolution time & packet sizes

DNS Transports
- UDP
- DTLSv1.2
- CoAPSv1.2
- OSCORE

CoAP Methods
- POST
- GET
- FETCH

A record
AAAA record

CDF
Resolution time [s]
Packet size L2 max. frame size

Frame Size [bytes]

Query [F/P] Query [G] Resp. (A) Resp. (AAAA)

UDP
CoAP
DTLSv1.2
CoAPSv1.2
OSCORE

0 32 64 96 128 160 192
0 32 64 96 128 160 192
0 32 64 96 128 160 192
0 32 64 96 128 160 192
0 32 64 96 128 160 192
0 32 64 96 128 160 192
0 32 64 96 128 160 192
0 32 64 96 128 160 192
0 32 64 96 128 160 192
0 32 64 96 128 160 192

UDP
CoAP
DTLSv1.2
CoAPSv1.2
OSCORE

Resolution time [s]
Packet size L2 max. frame size

Frame Size [bytes]
Experiment: Resolution time & packet sizes

**DNS Transports**
- **UDP**

**CoAP Methods**

**A record**

**AAAA record**

**CDF**

- Frame Size [bytes]

<table>
<thead>
<tr>
<th>Frame Size [bytes]</th>
<th>Resolution time [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>32</td>
<td>0.5</td>
</tr>
<tr>
<td>64</td>
<td>1.0</td>
</tr>
<tr>
<td>96</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td></td>
</tr>
<tr>
<td>160</td>
<td></td>
</tr>
<tr>
<td>192</td>
<td></td>
</tr>
</tbody>
</table>

**Packet size vs. L2 max. frame size**

- **UDP**

**Group 1**

No message fragmentation
Experiment: Resolution time & packet sizes

DNS Transports
- UDP
- CoAP

CoAP Methods
- POST
- FETCH

A record

AAAA record

CDF

Resolution time [s]

Packet size

Frame Size [bytes]

UDP

CoAP

Query [F/P]

Query [G]

Resp. (A)

Resp. (AAAA)

Group 2
Query unfragmented
Response fragmented
Experiment: Resolution time & packet sizes

Group 3
Both messages fragmented
Experiment: Resolution time & packet sizes

⇒ Fragmentation has larger impact on performance compared to transport or CoAP method
Problem:
Realistic query and response sizes lead to fragmentation, using OSCORE & 802.15.4

<table>
<thead>
<tr>
<th>Frame Size [bytes]</th>
<th>Name length = 2 chars (min)</th>
<th>Name length = 24 chars (median)</th>
<th>Name length = 25.9 chars (mean)</th>
<th>Name length = 83 chars (max)</th>
</tr>
</thead>
</table>
A New Content-Format: Numerical analysis

Problem:
Realistic query and response sizes lead to fragmentation, using OSCORE & 802.15.4

<table>
<thead>
<tr>
<th>Frame Size [bytes]</th>
<th>IEEE 802.15.4+6LoWPAN RIOT-most (w/o L2 security)</th>
<th>CoAP with OSCORE, Content-Format and URI-Path “/dns”</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>64</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td></td>
<td></td>
</tr>
<tr>
<td>192</td>
<td></td>
<td></td>
</tr>
<tr>
<td>256</td>
<td></td>
<td></td>
</tr>
<tr>
<td>320</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Reduce packet size via compression
Problem:
Realistic query and response sizes lead to fragmentation, using OSCORE & 802.15.4

IEEE 802.15.4+6LoWPAN RIOT-most (w/o L2 security)
CoAP with OSCORE, Content-Format and URI-Path “/dns”

IEEE 802.15.4+6LoWPAN RIOT-most (w/ L2 security)
CoAP with OSCORE, Content-Format and URI-Path “/dns”

⇒ Reduce packet size via compression
A New Content-Format: Some ideas

**Goal:** Reduce packet size

**Idea:**

- Omit authority and additional sections in DNS responses
- Question section always size 1: omit QDCount field
- Make class and type optional (imply IN/AAAA)
- Self-delimiting numeric values for classes, types, counts, TTLs, etc?
- Question section optional in responses?

**Two Options:**

- Question section CBOR-array, Answer section: CBOR-array of arrays?
- “remote getaddrbyname()” (i.e. query name (maybe type?), expect address as response)?

Discuss in separate draft?
Discussion: Caching and Max-Age vs. DNS TTL

**Problem:** CoAP Max-Age and DNS TTL may get out of sync at caching proxy

**Option 1** (PR#17): Do it like DoH but

**Server:**
Max-Age = \min(TTLs)

**Client:**
\[TTL_{new} = TTL_{old} - (\min(TTLs) - \text{Max-Age})\]

**Option 2** (PR#19): Do it like DoH but

**Server:**
Max-Age = \min(TTLs)
\[TTL_{new} = TTL_{old} - \min(TTLs)\]

**Client:**
\[TTL_{new} = TTL_{old} + \text{Max-Age}\]

(see GitHub-Issue #5)
Caching and Max-Age vs DNS TTL (Option 1, DoH-like)

DoC Client -> CoAP Proxy -> DoC Server

DoC query

Mostly trying to stay compatible with DoH
Caching and Max-Age vs DNS TTL (Option 1, DoH-like)

DoC Client → CoAP Proxy → DoC Server

DoC query → example.org,IN,AAAA

Mostly trying to stay compatible with DoH
Caching and Max-Age vs DNS TTL (Option 1, DoH-like)

DoC Client → CoAP Proxy → DoC Server

DoC query → DoC query

example.org, IN, AAAA
2001:db8::1, TTL=300
2001:db8::2, TTL=200

Mostly trying to stay compatible with DoH
Caching and Max-Age vs DNS TTL (Option 1, DoH-like)

DoC Client → CoAP Proxy → DoC Server

- DoC query
- DoC query

Example:
- Query to example.org, IN, AAAA
- Response: 2001:db8::1, TTL=300, Max-Age=200
- Response: 2001:db8::2, TTL=200

Mostly trying to stay compatible with DoH.
Caching and Max-Age vs DNS TTL (Option 1, DoH-like)

- **DoC Client**
  - `DoC query` to **CoAP Proxy**

- **CoAP Proxy**
  - `DoC query` to **DoC Server**
  - `DoC response`
  - `2.05, Max-Age=200`
  - `example.org, IN, AAAA`
  - `2001:db8::1, TTL=300`
  - `2001:db8::2, TTL=200`

- **DoC Server**
  - `DoC response`
  - `DoC query` from **CoAP Proxy**
Caching and Max-Age vs DNS TTL (Option 1, DoH-like)

- **DoC Client**
  - `DoC query`
  - `DoC response`

- **CoAP Proxy**
  - `DoC query` to `DoC Server`
  - `DoC response` from `DoC Server`

- **DoC Server**
  - `DoC query` from `CoAP Proxy`
  - `DoC response` to `CoAP Proxy`

**Example Queries and Responses:**
- `example.org,IN,AAAA`
- `2001:db8::1,TTL=300`
- `2001:db8::2,TTL=200`
- `2.05,Max-Age=200`
Caching and Max-Age vs DNS TTL (Option 1, DoH-like)

DoC Client → CoAP Proxy → DoC Server

- DoC query
- DoC response
- 2.05, Max-Age=200
- example.org, IN, AAAA
- 2001:db8::1, TTL=300
- 2001:db8::2, TTL=200

Mostly trying to stay compatible with DoH.
Caching and Max-Age vs DNS TTL (Option 1, DoH-like)

Mostly trying to stay compatible with DoH

<table>
<thead>
<tr>
<th>IP Address</th>
<th>TTL</th>
<th>Max-Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>example.org</td>
<td>200</td>
<td>200</td>
</tr>
<tr>
<td>2001:db8::1</td>
<td>300</td>
<td></td>
</tr>
<tr>
<td>2001:db8::2</td>
<td>200</td>
<td></td>
</tr>
</tbody>
</table>

DoC Client → DoC query → DoC response

CoAP Proxy → DoC query → DoC response

DoC Server

- 2.05, Max-Age=200
- example.org, IN, AAAA
- 2001:db8::1, TTL=300
- 2001:db8::2, TTL=200
Caching and Max-Age vs DNS TTL (Option 1, DoH-like)

DoC Client -> CoAP Proxy -> DoC Server

DoC query

DoC response

DoC query

DoC response

2.05, Max-Age=200

example.org,IN,AAAA

2001:db8::1,TTL=300

2001:db8::2,TTL=200

Mostly trying to stay compatible with DoH
Caching and Max-Age vs DNS TTL (Option 1, DoH-like)

DoC Client

DoC query

DoC response

DoC query

DoC response

CoAP Proxy

DoC query

DoC response

DoC query

example.org,IN,AAAA

2001:db8::1,TTL=300

2001:db8::2,TTL=200

DoC Server

2.05,Max-Age=200

Mostly trying to stay compatible with DoH
Caching and Max-Age vs DNS TTL (Option 1, DoH-like)

DoC Client  ->  CoAP Proxy  ->  DoC Server

DoC query  ->  DoC query  ->  DoC query

DoC response  ->  DoC response  ->  DoC response

---

**DoC query**

example.org,IN,AAAA

2001:db8::1,TTL=300
2001:db8::2,TTL=200

**DoC response**

2.05,Max-Age=200
example.org,IN,AAAA
2001:db8::1,TTL=300
2001:db8::2,TTL=200

---

Mostly trying to stay compatible with DoH.
Caching and Max-Age vs DNS TTL (Option 1, DoH-like)

DoC Client

CoAP Proxy

DoC Server

<table>
<thead>
<tr>
<th>2.05, Max-Age=200</th>
</tr>
</thead>
<tbody>
<tr>
<td>example.org, IN, AAAA</td>
</tr>
<tr>
<td>2001:db8::1, TTL=300</td>
</tr>
<tr>
<td>2001:db8::2, TTL=200</td>
</tr>
</tbody>
</table>

2.05, Max-Age=109
example.org, IN, AAAA
2001:db8::1, TTL=209
2001:db8::2, TTL=109

Mostly trying to stay compatible with DoH
Caching and Max-Age vs DNS TTL (Option 1, DoH-like)

Mostly trying to stay compatible with DoH
Caching and Max-Age vs DNS TTL (Option 2, adapt TTLs)

DoC Client → DoC query → CoAP Proxy → DoC query → DoC Server

example.org,IN,AAAA
2001:db8::1,TTL=300
2001:db8::2,TTL=200

Workload mostly at server + less cache invalidation
Caching and Max-Age vs DNS TTL (Option 2, adapt TTLs)

DoC Client → CoAP Proxy → DoC Server

DoC query → DoC query

<table>
<thead>
<tr>
<th>example.org,IN,AAAA</th>
</tr>
</thead>
<tbody>
<tr>
<td>2001:db8::1, TTL=300</td>
</tr>
<tr>
<td>2001:db8::2, TTL=200</td>
</tr>
</tbody>
</table>

Workload mostly at server + less cache invalidation
Caching and Max-Age vs DNS TTL (Option 2, adapt TTLs)

DoC Client  CoAP Proxy  DoC Server

DoC query  DoC query

example.org,IN,AAAA
2001:db8::1,TTL=100
2001:db8::2,TTL=0

2.05,Max-Age=200

Workload mostly at server + less cache invalidation
Caching and Max-Age vs DNS TTL (Option 2, adapt TTLs)

DoC Client

DoC query

DoC response

DoC query

DoC response

CoAP Proxy

DoC query

DoC response

DoC query

DoC response

DoC Server

2.05, Max-Age=200

domain.org, IN, AAAA

2001:db8::1, TTL=100

2001:db8::2, TTL=0

Workload mostly at server + less cache invalidation
Caching and Max-Age vs DNS TTL (Option 2, adapt TTLs)

DoC Client → CoAP Proxy → DoC Server

DoC query → DoC query → DoC query

DoC response → DoC response → DoC response

[example.org,IN,AAAA]
[2001:db8::1,TTL=100]
[2001:db8::2,TTL=0]

2.05,Max-Age=200

2.05,Max-Age=109

example.org,IN,AAAA

2001:db8::1,TTL=100
2001:db8::2,TTL=0
Caching and Max-Age vs DNS TTL (Option 2, adapt TTLs)

DoC Client

DoC query

CoAP Proxy

DoC query

DoC query

DoC response

DoC response

DoC response

DoC response

DoC Server

2.05, Max-Age=200
example.org, IN, AAAA
2001:db8::1, TTL=100
2001:db8::2, TTL=0

2.05, Max-Age=109
example.org, IN, AAAA
2001:db8::1, TTL=209
2001:db8::2, TTL=109

Workload mostly at server + less cache invalidation
Do we need to account for OBSERVE/Server Push?

Section 5.3:

- RFC 8484 (DoH), section 4.3: considerations on HTTP/2 Server Push
  - Deliver potential next request (e.g., website for queried domain name) to client together with DNS response
  - With CoRE: e.g., deliver .well-known/core content of CoRE-RD?
    - Requires CoAP request info in notification for proper caching
- Other use case for OBSERVE: RFC 8490, DNS Stateful Operations?
Issue #18 by Klaus Hartke proposes

• Specify REST API to retrieve DNS information from CoAP server instead
• Leave protocol details to implementation
Backup slides
Packet sizes by layer

IEEE 802.15.4+6LoWPAN RIOT-most (w/o L2 security)
CoAP with OSCORE, Content-Format and URI-Path “/dns”

IEEE 802.15.4+6LoWPAN RIOT-most (w/ L2 security)
CoAP with OSCORE, Content-Format and URI-Path “/dns”
Packet sizes: Best case L2 headers

Name length = 2 chars (min)

IEEE 802.15.4+6LoWPAN best header (w/o L2 security)
CoAP with OSCORE, Content-Format and URI-Path “/dns”

Name length = 24 chars (median)

IEEE 802.15.4+6LoWPAN best header (w/ L2 security, 16-bit key)
CoAP with OSCORE, Content-Format and URI-Path “/dns”

Name length = 25.9 chars (mean)

Name length = 83 chars (max)
Block-wise transfer

- RFC 7959 only
- Not yet looked into RFC 9177