DNS Queries over CoAP (DoC)
draft-lenders-dns-over-coap
(https://github.com/anr-bmbf-pivot/draft-dns-over-coap)

Martine S. Lenders, Christian Amsüss, Cenk Gündoğan,
Thomas C. Schmidt, Matthias Wählisch
IETF CoRE WG Interim Meeting, 2021-10-13
Outline

Introduction

Basic Message Exchange

Discussion
Motivation

Attack Scenario

Countermeasure

- Encrypt name resolution triggered by IoT devices against eavesdropping
- For the record, DoH and DoT conflict with constrained resources ;)
- Share system resources by using DNS over CoAP
  - Same socket and buffers can be used
  - Re-use of the CoAP retransmission mechanism
Overview

- GET coaps://[2001:db8::1]/?dns=example.org

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

/ CoAP request

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

[DNS response]
Message Formats

DNS Queries

• SHOULD be CoAP Confirmable (CON) message
• CoAP method types:
  1. GET: DNS query encoded in URI query parameter dns in "base64url"
  2. POST: DNS query in CoAP body as "application/dns-message"
  3. FETCH: DNS query in CoAP body as "application/dns-message"

DNS Responses

• DNS response in CoAP body of response
• DNS layer errors (e.g. SERVFAIL) SHOULD be carried in "2.xx Success" response
Caching and Timing

Problem

- Relative times control cache age in DNS (TTL) and CoAP (Max-Age)

Three Solutions

1. DoH way: use minimum TTL for Max-Age
   ⇒ Relative time frames make TTL go out of sync when cached
2. Adopt Max-Age to TTL (or vice-versa): Clients calculate actual TTL from Max-Age
3. New (absolute) Age option as in HTTP?

Details

- https://github.com/anr-bmbf-pivot/draft-dns-over-coap/issues/5
Do we really need GET, POST, and FETCH?

<table>
<thead>
<tr>
<th></th>
<th>GET</th>
<th>POST</th>
<th>FETCH</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cacheable</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
</tr>
<tr>
<td>Block-wise transferable query</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>URI Template not needed</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

Current State of Discussion

- Solution w/o URI Template eases integration into CoRE-RD
- But, if FETCH is not widely deployed, “Wild” implementations using GET or POST may pop up
- Compromise: Define GET/POST usage in Appendix

Details

- https://github.com/anr-bmbf-pivot/draft-dns-over-coap/pull/8
FETCH Implementation Status

FETCH supported in 9 of 15 active FOSS implementations (last commit <1y):

<table>
<thead>
<tr>
<th>aiocoap</th>
<th>Californium</th>
<th>cantcoap</th>
<th>coap-lite</th>
<th>Erbium (Wakaama)</th>
<th>gcoap (RIOT)</th>
<th>Go-CoAP (go-oct)</th>
<th>libcoap</th>
<th>Lobar CoAP</th>
<th>mbed-CoAP</th>
<th>nanocoap (RIOT)</th>
<th>node-coap</th>
<th>Waher.Networking.CoAP</th>
<th>Zephyr CoAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✗</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✔️</td>
<td>✗</td>
</tr>
</tbody>
</table>

+ FETCH implementation is near trivial

Full overview: https://gist.github.com/miri64/48d83316afd1d5e600ed2ec0bbab3624
Discussion Topics

1. Is this topic relevant for CoRE?
2. How to handle relative aging and caching?
3. Should we abandon GET and POST in DoC completely or partly?
4. ...