Stream: Internet Engineering Task Force (IETF)
RFC: 9953
Category: Standards Track
Published: February 2026
ISSN: 2070-1721
Authors:
M. S. Lenders, *TU Dresden*
C. Amsüss
C. Gündoğan, *NeuralAgent GmbH*
T. C. Schmidt, *HAW Hamburg*
M. Wählisch, *TU Dresden & Barkhausen Institut*

RFC 9953
DNS over the Constrained Application Protocol (DoC)

Abstract

This document defines a protocol for exchanging DNS queries (OPCODE 0) over the Constrained Application Protocol (CoAP). These CoAP messages can be protected by (D)TLS-Secured CoAP (CoAPS) or Object Security for Constrained RESTful Environments (OSCORE) to provide encrypted DNS message exchange for constrained devices in the Internet of Things (IoT).

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9953.

Table of Contents

1. Introduction

This document defines DNS over CoAP (DoC), a protocol to send DNS [STD13] queries and get DNS responses over the Constrained Application Protocol (CoAP) [RFC7252] using OPCODE 0 (Query). Each DNS query-response pair is mapped into a CoAP message exchange. Each CoAP message can be secured by DTLS 1.2 or newer [RFC6347] [RFC9147] as well as Object Security for Constrained RESTful Environments (OSCORE) [RFC8613] and TLS 1.3 or newer [RFC8323] [RFC8446] to ensure message integrity and confidentiality.

The application use case of DoC is inspired by DNS over HTTPS (DoH) [RFC8484]. However, DoC aims for deployment in the constrained Internet of Things (IoT), which usually conflicts with the requirements introduced by HTTPS. Constrained IoT devices may be restricted in memory, power consumption, link-layer frame sizes, throughput, and latency. They may only have a handful kilobytes of both RAM and ROM. They may sleep for long durations of time, after which they need to refresh the named resources they know about. Name resolution in such scenarios must take into account link-layer frame sizes of only a few hundred bytes, bit rates in the magnitude of kilobits per second, and latencies of several seconds [RFC7228] [CONSTR-NODES-BIS].

In order not to be burdened by the resource requirements of TCP and HTTPS, constrained IoT devices could use DNS over DTLS [RFC8094]. In contrast to DNS over DTLS, DoC can take advantage of CoAP features to mitigate drawbacks of datagram-based communication. These features include: block-wise transfer [RFC7959], which solves the Path MTU problem of DNS over

DTLS (see [RFC8094], Section 5); CoAP proxies, which provide an additional level of caching; and reuse of data structures for application traffic and DNS information, which saves memory on constrained devices.

To avoid the resource requirements of DTLS or TLS on top of UDP (e.g., introduced by DNS over DTLS [RFC8094] or DNS over QUIC [RFC9250]), DoC allows for lightweight message protection based on OSCORE.

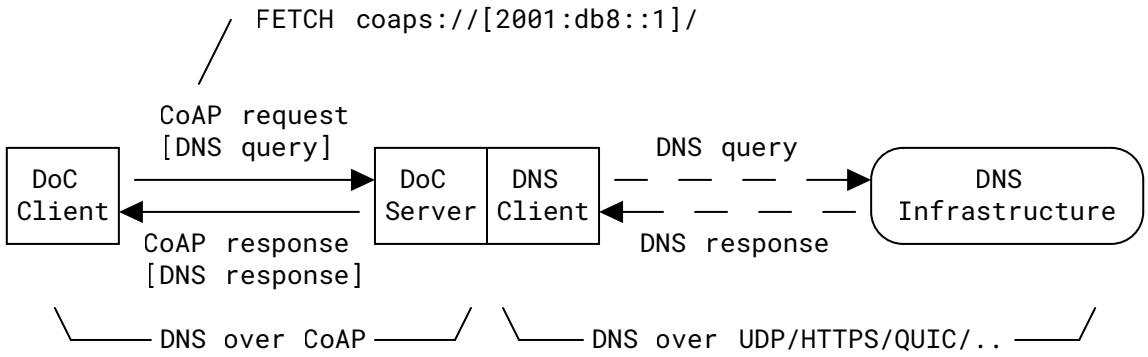

Figure 1: Basic DoC Architecture

The most important components of DoC can be seen in Figure 1: a DoC client tries to resolve DNS information by sending DNS queries carried within CoAP requests to a DoC server. That DoC server can be the authoritive name server for the queried record or a DNS client (i.e., a stub or recursive resolver) that resolves DNS information by using other DNS transports such as DNS over UDP [STD13], DNS over HTTPS [RFC8484], or DNS over QUIC [RFC9250] when communicating with the upstream DNS infrastructure. Using that information, the DoC server then replies to the queries of the DoC client with DNS responses carried within CoAP responses. A DoC server **MAY** also serve as a DNSSEC validator to provide DNSSEC validation to the more constrained DoC clients.

Note that this specification is distinct from DoH because the CoAP-specific FETCH method [RFC8132] is used. A benefit of using this method is having the DNS query in the body such as when using the POST method, but with the same caching advantages of responses to requests that use the GET method. Having the DNS query in the body means that there is no need for extra base64 encoding, which would increase code complexity and message sizes. Also, this allows for the block-wise transfer of queries [RFC7959].

2. Terminology and Conventions

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

A server that provides the service specified in this document is called a "DoC server" to differentiate it from a classic "DNS server". A DoC server acts as either a DNS stub resolver or a DNS recursive resolver [BCP219]. As such, the DoC server communicates with an "upstream DNS infrastructure" or a single "upstream DNS server". A "DoC resource" is a CoAP resource [RFC7252] at the DoC server that DoC clients can target in order to send a DNS query in a CoAP request.

A client using the service specified in this document to retrieve the DNS information is called a "DoC client".

The term "constrained nodes" is used as defined in [RFC7228]. [RFC6690] describes that Constrained RESTful Environments (CoRE) realize the Representational State Transfer (REST) architecture [REST] in a suitable form for such constrained nodes.

A DoC server can provide Observe capabilities as defined in [RFC7641], Section 1.2. As part of that, it administers a "list of observers". DoC clients using these capabilities are "observers" as defined in [RFC7641], Section 1.2. A "notification" is a CoAP response message with an Observe option; see [RFC7641], Section 4.2.

The terms "payload" and "body" are used as defined in [RFC7959], Section 2. Note that, when block-wise transfer is not used, the terms "payload" and "body" are to be understood as equal.

In the examples in this document, the binary payload and resource records are shown in a hexadecimal representation as well as a human-readable format for better readability. However, in the actual message sent and received, they are encoded in the binary message format defined in [STD13].

3. Selection of a DoC Server

While there is no path specified for the DoC resource, it is **RECOMMENDED** to use the root path "/" to keep the CoAP requests small.

The DoC client needs to know the DoC server and the DoC resource at the DoC server. Possible options to assure this could be (1) manual configuration of a Uniform Resource Identifier (URI) [RFC3986] or Constrained Resource Identifier (CRI) [CRI] or (2) automatic configuration, e.g., using a CoRE resource directory [RFC9176], DHCP or Router Advertisement options [RFC9463], or discovery of designated resolvers [RFC9462]. Automatic configuration **MUST** only be done from a trusted source.

3.1. Discovery by Resource Type

For discovery of the DoC resource through a link mechanism that allows describing a resource type (e.g., the Resource Type attribute in [RFC6690]), this document introduces the resource type "core.dns". It can be used to identify a generic DNS resolver that is available to the client.

3.2. Discovery Using SVCB Resource Records or DNR

A DoC server can also be discovered using Service Binding (SVCB) Resource Records (RRs) [RFC9460] [RFC9461] resolved via another DNS service (e.g., provided by an unencrypted local resolver) or Discovery of Network-designated Resolvers (DNR) Service Parameters [RFC9463] via DHCP or Router Advertisements. [RFC8323] defines the Application-Layer Protocol Negotiation (ALPN) ID for CoAP over TLS servers and [PRE-RFC9952] defines the ALPN ID for CoAP over DTLS servers. DoC servers that use only OSCORE [RFC8613] and Ephemeral Diffie-Hellman Over COSE (EDHOC) [RFC9528] (COSE stands for "Concise Binary Object Notation (CBOR) Object Signing and Encryption" [RFC9052]) to support security cannot be discovered using these SVCB RR or DNR mechanisms. Specifying an alternate discovery mechanism is out of the scope of this document.

This document is not an SVCB mapping document for the CoAP schemes as defined in Section 2.4.3 of [RFC9460]. A full SVCB mapping is specified in [TRANSPORT-IND]. It generalizes mechanisms for all CoAP services. This document introduces only the discovery of DoC services.

This document specifies "docpath" as a single-valued Service Parameter Key (SvcParamKey) that is mandatory for DoC SVCB records. If the "docpath" SvcParamKey is absent, the service should not be considered a valid DoC service.

The docpath is divided up into segments of the absolute path to the DoC resource (docpath-segment), each a sequence of 1-255 octets. In ABNF [RFC5234]:

```
docpath-segment = 1*255OCTET
```

Note that this restricts the length of each docpath-segment to at most 255 octets. Paths with longer segments cannot be advertised with the "docpath" SvcParam and are thus **NOT RECOMMENDED** for the path to the DoC resource.

The presentation format value of "docpath" **SHALL** be a comma-separated list (Appendix A.1 of [RFC9460]) of 0 or more docpath-segments. The root path "/" is represented by 0 docpath-segments, i.e., an empty list. The same considerations apply for the "," and "" characters in docpath-segments for zone-file implementations and the alpn-ids in an "alpn" SvcParam (Section 7.1.1 of [RFC9460]).

The wire-format value for "docpath" consists of 0 or more sequences of octets prefixed by their respective length as a single octet. We call this single octet the length octet. The length octet and the corresponding sequence form a length-value pair. These length-value pairs are concatenated to form the SvcParamValue. These pairs **MUST** exactly fill the SvcParamValue; otherwise, the SvcParamValue is malformed. Each such length-value pair represents one segment of the absolute path to the DoC resource. The root path "/" is represented by 0 length-value pairs, i.e., SvcParamValue length 0.

Note that this format uses the same encoding as the "alpn" SvcParam, and it can reuse the decoders and encoders for that SvcParam with the adaption that a length of zero is allowed. As long as each docpath-segment is a length 0 and 24 octets, it is easily transferred into the path representation in CRIs [CRI] by masking each length octet with the CBOR text string major type 3 (`0x60` as an octet; see [RFC8949]). Furthermore, it is easily transferable into a sequence of CoAP Uri-Path options by mapping each length octet into the Option Delta and Option Length of the corresponding CoAP Uri-Path option, provided the docpath-segments are all of a length between 0 and 12 octets (see [RFC7252], Section 3.1). Likewise, it can be transferred into a URI path-abempty form by replacing each length octet with the "/" character None of the abovementioned prevent longer docpath-segments than the considered, they just make the translation harder, as they require to make space for the longer delimiters, in turn requiring to move octets.

To use the service binding from an SVCB RR or DNR Encrypted DNS option, the DoC client **MUST** send a DoC request constructed from the SvcParams including "docpath". The construction algorithm for DoC requests is as follows, going through the provided records in order of their priority. For the purposes of this algorithm, the DoC client is assumed to be SVCB-optional (see Section 3 of [RFC9460]).

- If the "alpn" SvcParam value for the service is "coap", a CoAP request for CoAP over TLS **MUST** be constructed [RFC8323]. If it is "co", a CoAP request for CoAP over DTLS **MUST** be constructed [PRE-RFC9952]. Any other SvcParamKeys specifying a transport are out of the scope of this document.
- The destination address for the request **SHOULD** be taken from additional information about the target. This may include (1) A or AAAA RRs associated with the target name and delivered with the SVCB RR (see [RFC9462]), (2) "ipv4hint" or "ipv6hint" SvcParams from the SVCB RR (see [RFC9461]), or (3) from IPv4 or IPv6 addresses provided if DNR [RFC9463] is used. As a fallback, an address **MAY** be queried for the target name of the SVCB record from another DNS service.
- The destination port for the request **MUST** be taken from the "port" SvcParam value, if present. Otherwise, take the default port of the CoAP transport, e.g., with regards to this specification, the default is TCP port 5684 for "coap" or UDP port 5684 for "co". This document introduces no limitations on the ports that can be used. If a malicious SVCB record allows its originator to influence message payloads, Section 12 of [RFC9460] recommends placing such restrictions in the SVCB mapping document. The records used in this document only influence the Uri-Path option. That option is only sent in the plaintext of an encrypted (D)TLS channel and thus does not warrant any limitations.
- The request URI's hostname component **MUST** be the Authentication Domain Name (ADN) when obtained through DNR and **MUST** be the target name of the SVCB record when obtained through a `_dns` query (if AliasMode is used to the target name of the AliasMode record). This can be achieved efficiently by setting that name in TLS Server Name Indication (SNI) [RFC8446] or by setting the Uri-Host option on each request.
- For each element in the CBOR sequence of the "docpath" SvcParam value, a Uri-Path option **MUST** be added to the request.

- If the request constructed this way receives a response, the same SVCB record **MUST** be used for construction of future DoC queries. If not, or if the endpoint becomes unreachable, the algorithm repeats with the SVCB RR or DNR Encrypted DNS option with the next highest Service Priority as a fallback (see Sections 2.4.1 and 3 of [RFC9460]).

A more generalized construction algorithm for any CoAP request can be found in [TRANSPORT-IND].

3.2.1. Examples

A typical SVCB resource record response for a DoC server at the root path "/" of the server looks like the following (the "docpath" SvcParam is the last 4 bytes `00 0a 00 00` in the binary):

```
Resource record (binary):
  04 5f 64 6e 73 07 65 78 61 6d 70 6c 65 03 6f 72
  67 00 00 40 00 01 00 00 06 28 00 1e 00 01 03 64
  6e 73 07 65 78 61 6d 70 6c 65 03 6f 72 67 00 00
  01 00 03 02 63 6f 00 0a 00 00

Resource record (human-readable):
  _dns.example.org.  1576  IN SVCB 1 dns.example.org (
      alpn=co docpath )
```

The root path is **RECOMMENDED** but not required. Here are two examples where the "docpath" represents paths of varying depth. First, "/dns" is provided in the following example (the last 8 bytes `00 0a 00 04 03 64 6e 73`):

```
Resource record (binary):
  04 5f 64 6e 73 07 65 78 61 6d 70 6c 65 03 6f 72
  67 00 00 40 00 01 00 00 00 55 00 22 00 01 03 64
  6e 73 07 65 78 61 6d 70 6c 65 03 6f 72 67 00 00
  01 00 03 02 63 6f 00 0a 00 04 03 64 6e 73

Resource record (human-readable):
  _dns.example.org.    85  IN SVCB 1 dns.example.org (
      alpn=co docpath=dns )
```

Second, see an example for the path "/n/s" (the last 8 bytes `00 0a 00 04 01 6e 01 73`):

```
Resource record (binary):
  04 5f 64 6e 73 07 65 78 61 6d 70 6c 65 03 6f 72
  67 00 00 40 00 01 00 00 06 6b 00 22 00 01 03 64
  6e 73 07 65 78 61 6d 70 6c 65 03 6f 72 67 00 00
  01 00 03 02 63 6f 00 0a 00 04 01 6e 01 73

Resource record (human-readable):
  _dns.example.org.   643  IN SVCB 1 dns.example.org (
      alpn=co docpath=n,s )
```

If the server also provides DNS over HTTPS, "dohpath" and "docpath" **MAY** coexist:

```
Resource record (binary):
  04 5f 64 6e 73 07 65 78 61 6d 70 6c 65 03 6f 72
  67 00 00 40 00 01 00 00 01 ad 00 2b 00 01 03 64
  6e 73 07 65 78 61 6d 70 6c 65 03 6f 72 67 00 00
  01 00 06 02 68 33 02 63 6f 00 07 00 07 2f 7b 3f
  64 6e 73 7d 00 0a 00 00

Resource record (human-readable):
  _dns.example.org.   429  IN SVCB 1 dns.example.org (
      alpn=h3,co dohpath=/{?dns} docpath )
```

4. Basic Message Exchange

4.1. The "application/dns-message" Content-Format

This document defines a CoAP Content-Format ID for the Internet media type "application/dns-message" to be the mnemonic 553, based on the port assignment of DNS. This media type is defined as in Section 6 of [RFC8484], i.e., a single DNS message encoded in the DNS on-the-wire format [STD13]. Both DoC client and DoC server **MUST** be able to parse contents in the "application/dns-message" Content-Format. This document only specifies OPCODE 0 (Query) for DNS over CoAP messages. Future documents can provide considerations for additional OPCODEs or extend its specification (e.g., by describing whether other CoAP codes need to be used for which OPCODE). Unless another error takes precedence, a DoC server uses RCODE = 4, NotImp [STD13], in its response to a query with an OPCODE that it does not implement (see also Section 4.3.3).

4.2. DNS Queries in CoAP Requests

A DoC client encodes a single DNS query in one or more CoAP request messages that use the CoAP FETCH [RFC8132] request method. Requests **SHOULD** include an Accept option to indicate the type of content that can be parsed in the response.

Since CoAP provides reliability at the message layer (e.g., through Confirmable messages), the retransmission mechanism of the DNS protocol as defined in [STD13] is not needed.

4.2.1. Request Format

When sending a CoAP request, a DoC client **MUST** include the DNS query in the body of the CoAP request. As specified in Section 2.3.1 of [RFC8132], the type of content of the body **MUST** be indicated using the Content-Format option. This document specifies the usage of Content-Format "application/dns-message" (for details, see Section 4.1).

4.2.2. Support of CoAP Caching

The DoC client **SHOULD** set the ID field of the DNS header to 0 to enable a CoAP cache (e.g., a CoAP proxy en route) to respond to the same DNS queries with a cache entry. This ensures that the CoAP Cache-Key (see [RFC8132], Section 2) does not change when multiple DNS queries for the same DNS data, carried in CoAP requests, are issued. Apart from losing these caching

benefits, there is no harm in not setting it to 0, e.g., when the query was received from somewhere else. In any instance, a DoC server **MUST** copy the ID from the query in its response to that query.

4.2.3. Example

The following example illustrates the usage of a CoAP message to resolve "example.org. IN AAAA" based on the URI "coaps://[2001:db8::1]/". The CoAP body is encoded in the "application/dns-message" Content-Format.

```
FETCH coaps://[2001:db8::1]/
Content-Format: 553 (application/dns-message)
Accept: 553 (application/dns-message)
Payload (binary):
  00 00 01 00 00 01 00 00 00 00 00 00 07 65 78 61
  6d 70 6c 65 03 6f 72 67 00 00 1c 00 01

Payload (human-readable):
  ;; ->>Header<<- opcode: QUERY, status: NOERROR, id: 0
  ;; flags: rd; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0

  ;; QUESTION SECTION:
  ;example.org.             IN      AAAA
```

4.3. DNS Responses in CoAP Responses

Each DNS query-response pair is mapped to a CoAP request-response operation. DNS responses are provided in the body of the CoAP response, i.e., it is also possible to transfer them using block-wise transfer [RFC7959]. A DoC server **MUST** be able to produce responses in the "application/dns-message" Content-Format (for details, see Section 4.1) when requested. The use of the Accept option in the request is optional. However, all DoC clients **MUST** be able to parse an "application/dns-message" response (see also Section 4.1). Any response Content-Format other than "application/dns-message" **MUST** be indicated with the Content-Format option by the DoC server.

4.3.1. Response Codes and Handling DNS and CoAP Errors

A DNS response indicates either success or failure in the RCODE of the DNS header (see [STD13]). It is **RECOMMENDED** that CoAP responses that carry a parsable DNS response use a 2.05 (Content) response code.

CoAP responses using non-successful response codes **MUST NOT** contain a DNS response and **MUST** only be used for errors in the CoAP layer or when a request does not fulfill the requirements of the DoC protocol.

Communication errors with an upstream DNS server (e.g., timeouts) **MUST** be indicated by including a DNS response with the appropriate RCODE in a successful CoAP response, i.e., using a 2.xx response code. When an error occurs at the CoAP layer, e.g., if an unexpected request method or an unsupported Content-Format in the request are used, the DoC server **SHOULD** respond with an appropriate CoAP error.

A DoC client might try to repeat a non-successful exchange unless otherwise prohibited. The DoC client might also decide to repeat a non-successful exchange with a different URI, for instance, when the response indicates an unsupported Content-Format.

4.3.2. Support of CoAP Caching

For reliability and energy-saving measures, content decoupling (such as en-route caching on proxies) takes a far greater role than it does in HTTP. Likewise, CoAP makes it possible to use cache validation to refresh stale cache entries to reduce the number of large response messages. For cache validation, CoAP implementations regularly use hashing over the message content for ETag generation (see [RFC7252], Section 5.10.6). As such, the approach to guarantee the same cache key for DNS responses as proposed in DoH ([RFC8484], Section 5.1) is not sufficient and needs to be updated so that the TTLs in the response are more often the same regardless of query time.

The DoC server **MUST** ensure that the sum of the Max-Age value of a CoAP response and any TTL in the DNS response is less than or equal to the corresponding TTL received from an upstream DNS server. This also includes the default Max-Age value of 60 seconds (see Section 5.10.5 of [RFC7252]) when no Max-Age option is provided. The DoC client **MUST** then add the Max-Age value of the carrying CoAP response to all TTLs in a DNS response on reception and use these calculated TTLs for the associated records.

To meet the requirement for DoC, the **RECOMMENDED** algorithm for a DoC server is as follows: Set the Max-Age option of a response to the minimum TTL of a DNS response and subtract this value from all TTLs of that DNS response. This prevents expired records from unintentionally being served from an intermediate CoAP cache. Additionally, if the ETag for cache validation is based on the content of the response, it allows that ETag not to change. This then remains the case even if the TTL values are updated by an upstream DNS cache. If only one record set per DNS response is assumed, a simplification of this algorithm is to just set all TTLs in the response to 0 and set the TTLs at the DoC client to the value of the Max-Age option.

If shorter caching periods are plausible, e.g., if the RCODE of the message indicates an error that should only be cached for a minimal duration, the value for the Max-Age option **SHOULD** be set accordingly. This value might be 0, but if the DoC server knows that the error will persist, greater values are also conceivable, depending on the projected duration of the error. The same applies for DNS responses that, for any reason, do not carry any records with a TTL.

4.3.3. Examples

The following example illustrates the response to the query "example.org. IN AAAA record", with recursion turned on. Successful responses carry one answer record including the address 2001:db8:1:0:1:2:3:4 and TTL 79689.

A successful response:

```
2.05 Content
Content-Format: 553 (application/dns-message)
Max-Age: 58719
Payload (human-readable):
 ;; ->>Header<<- opcode: QUERY, status: NOERROR, id: 0
 ;; flags: qr rd ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 0

 ;; QUESTION SECTION:
 ;example.org.                IN      AAAA
 ;; ANSWER SECTION:
 ;example.org.         79689   IN      AAAA    2001:db8:1:0:1:2:3:4
```

When a DNS error -- NxDomain (RCODE = 3) for "does.not.exist" in this case -- is noted in the DNS response, the CoAP response still indicates success.

```
2.05 Content
Content-Format: 553 (application/dns-message)
Payload (human-readable):
 ;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 0
 ;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0

 ;; QUESTION SECTION:
 ;does.not.exist.              IN      AAAA
```

As described in Section 4.1, a DoC server uses NotImp (RCODE = 4) if it does not support an OPCODE-a DNS Update (OPCODE = 5) for "example.org" in this case.

```
2.05 Content
Content-Format: 553 (application/dns-message)
Payload (human-readable):
 ;; ->>Header<<- opcode: UPDATE, status: NOTIMP, id: 0
 ;; flags: qr ra; QUERY: 1, ANSWER: 0, AUTHORITY: 0, ADDITIONAL: 0

 ;; QUERY SECTION:
 ;example.org.                IN      AAAA
```

When an error occurs at the CoAP layer, the DoC server responds with an appropriate CoAP error, for instance, 4.15 (Unsupported Content-Format) if the Content-Format option in the request was not set to "application/dns-message" and the Content-Format is not otherwise supported by the server.

```
4.15 Unsupported Content-Format
[no payload]
```

5. Interaction with Other CoAP and CoRE Features

5.1. DNS Push Notifications and CoAP Observe

DNS Push Notifications [RFC8765] provide the capability to asynchronously notify clients about resource record changes. However, it results in additional overhead, which conflicts with constrained resources. This is the reason why it is **RECOMMENDED** to use CoAP Observe [RFC7641] instead of DNS Push in the DoC domain. This is particularly useful if a short-lived record is requested frequently. The DoC server **SHOULD** provide Observe capabilities on its DoC resource and do as follows.

If a DoC client wants to observe a resource record, a DoC server can respond with a notification and add the client to its list of observers for that resource in accordance with [RFC7641]. The DoC server **MAY** subscribe to DNS push notifications for that record. This involves sending a DNS Subscribe message (see Section 6.2 of [RFC8765]), instead of a classic DNS query to fetch the information on behalf of the DoC client.

After the list of observers for a particular DNS query has become empty (by explicit or implicit cancellation of the observation as per Section 3.6 of [RFC7641]), and no other reason to subscribe to that request is present, the DoC server **SHOULD** cancel the corresponding subscription. This can involve sending a DNS Unsubscribe message or closing the session (see Section 6.4 of [RFC8765]). As there is no CoAP observer anymore from the perspective of the DoC server, a failure to do so cannot be communicated back to any DoC observer. As such, error handling (if any) needs to be resolved between the DoC server and the upstream DNS infrastructure.

Whenever the DoC server receives a DNS Push message from the DNS infrastructure for an observed resource record, the DoC server sends an appropriate Observe notification response to the DoC client.

A server that responds with notifications (i.e., sends the Observe option) needs to have the means of obtaining current resource records. This may happen through DNS Push or also by upstream polling or implicit circumstances (e.g., if the DoC server is the authoritative name server for the record and wants to notify about changes).

5.2. OSCORE

It is **RECOMMENDED** to carry DNS messages protected using OSCORE [RFC8613] between the DoC client and the DoC server. The establishment and maintenance of the OSCORE security context is out of the scope of this document.

[CACHABLE-OSCORE] describes a method to allow cache retrieval of OSCORE responses and discusses the corresponding implications on message sizes and security properties.

5.3. Mapping DoC to DoH

This document provides no specification on how to map between DoC and DoH, e.g., at a CoAP-to-HTTP proxy. Such a direct mapping is **NOT RECOMMENDED**: Rewriting the FETCH method (Section 4.2) and TTL (Section 4.3.2) as specified in this document would be non-trivial. It is **RECOMMENDED** to use a DNS forwarder to map between DoC and DoH, as would be the case for mapping between any other pair of DNS transports.

6. Considerations for Unprotected Use

The use of DoC without confidentiality and integrity protection is **NOT RECOMMENDED**. Without secure communication, many possible attacks need to be evaluated in the context of the application's threat model. This includes known threats for unprotected DNS [RFC3833] [RFC9076] and CoAP (Section 11 of [RFC7252]). While DoC does not use the random ID of the DNS header (see Section 4.2.2), equivalent protection against off-path poisoning attacks is achieved by using random large token values for unprotected CoAP requests. If a DoC message is unprotected, it **MUST** use a random token with a length of at least 2 bytes to mitigate this kind of poisoning attack.

7. Security Considerations

General CoAP security considerations ([RFC7252], Section 11) apply to DoC. DoC also inherits the security considerations of the protocols used for secure communication, e.g., OSCORE ([RFC8613], Section 12) as well as DTLS 1.2 or newer ([RFC6347], Section 5 and [RFC9147], Section 11). Additionally, DoC uses request patterns that require the maintenance of long-lived security contexts. Section 2.6 of [CoAP-CORR-CLAR] provides insights on what can be done when those are resumed from a new endpoint.

Though DTLS v1.2 [RFC6347] was obsoleted by DTLS v1.3 [RFC9147], there are many CoAP implementations that still use v1.2 at the time of writing. As such, this document also accounts for the usage of DTLS v1.2 even though newer versions are **RECOMMENDED** when using DTLS to secure CoAP.

When using unprotected CoAP (see Section 6), setting the ID of a DNS message to 0 as specified in Section 4.2.2 opens the DNS cache of a DoC client to cache poisoning attacks via response spoofing. This document requires an unpredictable CoAP token in each DoC query from the client when CoAP is not secured to mitigate such an attack over DoC (see Section 6).

For secure communication via (D)TLS or OSCORE, an unpredictable ID to protect against spoofing is not necessary. Both (D)TLS and OSCORE offer mechanisms to harden against injecting spoofed responses in their protocol design. Consequently, the ID of the DNS message can be set to 0 without any concern in order to leverage the advantages of CoAP caching.

A DoC client must be aware that the DoC server may communicate unprotected with the upstream DNS infrastructure, e.g., using DNS over UDP. DoC can only guarantee confidentiality and integrity of communication between parties for which the security context is exchanged. The DoC server may use another security context to communicate upstream with both confidentiality and integrity (e.g., DNS over QUIC [RFC9250]); however, while recommended, this is opaque to the DoC client on the protocol level. Record integrity can also be ensured upstream using DNSSEC [BCP237].

A DoC client may not be able to perform DNSSEC validation, e.g., due to code size constraints or the size of the responses. It may trust its DoC server to perform DNSSEC validation; how that trust is expressed is out of the scope of this document. For instance, a DoC client may be configured to use a particular credential by which it recognizes an eligible DoC server. That information can also imply trust in the DNSSEC validation by that DoC server.

8. IANA Considerations

8.1. CoAP Content-Formats Registry

IANA has assigned a CoAP Content-Format ID for the "application/dns-message" media type in the "CoAP Content-Formats" registry in the "Constrained RESTful Environments (CoRE) Parameters" registry group [RFC7252]; this corresponds to the "application/dns-message" media type from the "Media Types" registry (see [RFC8484]).

Content Type	ID	Reference
application/dns-message	553	[RFC8484] and RFC 9953, Section 4.1

Table 1: CoAP Content-Format ID

8.2. DNS SVBC Service Parameter Keys (SvcParamKeys) Registry

IANA has added the following entry to the "DNS SVCB Service Parameter Keys (SvcParamKeys)" registry in the "DNS Service Bindings (SVCB)" registry group. The definition of this parameter can be found in Section 3.

Number	Name	Meaning	Change Controller	Reference
10	docpath	DNS over CoAP resource path	IETF	RFC 9953, Section 3

Table 2: Value for SvcParamKeys

8.3. Resource Type (rt=) Link Target Attribute Values Registry

IANA has added "core.dns" to the "Resource Type (rt=) Link Target Attribute Values" registry in the "Constrained RESTful Environments (CoRE) Parameters" registry group.

Value	Description	Reference
core.dns	DNS over CoAP resource	RFC 9953, Section 3

Table 3: Resource Type (rt=) Link Target Attribute

9. Operational Considerations

9.1. Coexistence of Different DNS and CoAP Transports

Many DNS transports may coexist on the DoC server, such as DNS over UDP [STD13], DNS over (D)TLS [RFC7858] [RFC8094], DNS over HTTPS [RFC8484], or DNS over QUIC [RFC9250]. In principle, transports employing channel or object security should be preferred. In constrained scenarios, DNS over CoAP is preferable to DNS over DTLS. The final decision regarding the preference, however, heavily depends on the use case and is therefore left to the implementers or users and is not defined in this document.

CoAP supports Confirmable and Non-Confirmable messages [RFC7252] to deploy different levels of reliability. However, this document does not enforce any of these message types, as the decision on which one is appropriate depends on the characteristics of the network where DoC is deployed.

9.2. Redirects

Application-layer redirects (e.g., HTTP) redirect a client to a new server. In the case of DoC, this leads to a new DNS server. This new DNS server may provide different answers to the same DNS query than the previous DNS server. At the time of writing, CoAP does not support redirection. Future specifications of CoAP redirect may need to consider the impact of different results between previous and new DNS servers.

9.3. Proxy Hop Limit

Mistakes might lead to CoAP proxies forming infinite loops. Using the CoAP Hop-Limit option [RFC8768] mitigates such loops.

9.4. Error Handling

Section 4.3.1 specifies that DNS operational errors should be reported back to a DoC client using the appropriate DNS RCODE. If a DoC client did not receive any successful DNS messages from a DoC server for a while, it might indicate that the DoC server lost connectivity to the upstream DNS infrastructure. The DoC client should handle this error case like a recursive resolver that lost connectivity to the upstream DNS infrastructure. In case of CoAP errors, the usual mechanisms for CoAP response codes apply.

9.5. DNS Extensions

DNS extensions that are specific to the choice of transport, such as described in [RFC7828], are not applicable to DoC.

10. References

10.1. Normative References

[PRE-RFC9952] Lenders, M. S., Amsüss, C., Schmidt, T. C., and M. Wählisch, "The Application-Layer Protocol Negotiation (ALPN) ID Specification for the Constrained Application Protocol (CoAP) over DTLS", RFC PRE-9952, DOI 10.17487/PRE-RFC9952, March 2026, <https://www.rfc-editor.org/info/rfc9952>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[RFC7641] Hartke, K., "Observing Resources in the Constrained Application Protocol (CoAP)", RFC 7641, DOI 10.17487/RFC7641, September 2015, <https://www.rfc-editor.org/info/rfc7641>.

[RFC7959] Bormann, C. and Z. Shelby, Ed., "Block-Wise Transfers in the Constrained Application Protocol (CoAP)", RFC 7959, DOI 10.17487/RFC7959, August 2016, <https://www.rfc-editor.org/info/rfc7959>.

[RFC8132] van der Stok, P., Bormann, C., and A. Sehgal, "PATCH and FETCH Methods for the Constrained Application Protocol (CoAP)", RFC 8132, DOI 10.17487/RFC8132, April 2017, <https://www.rfc-editor.org/info/rfc8132>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8323] Bormann, C., Lemay, S., Tschofenig, H., Hartke, K., Silverajan, B., and B. Raymor, Ed., "CoAP (Constrained Application Protocol) over TCP, TLS, and WebSockets", RFC 8323, DOI 10.17487/RFC8323, February 2018, <https://www.rfc-editor.org/info/rfc8323>.

[RFC8484] Hoffman, P. and P. McManus, "DNS Queries over HTTPS (DoH)", RFC 8484, DOI 10.17487/RFC8484, October 2018, <https://www.rfc-editor.org/info/rfc8484>.

[RFC8613] Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)", RFC 8613, DOI 10.17487/RFC8613, July 2019, <https://www.rfc-editor.org/info/rfc8613>.

[RFC8765] Pusateri, T. and S. Cheshire, "DNS Push Notifications", RFC 8765, DOI 10.17487/RFC8765, June 2020, <https://www.rfc-editor.org/info/rfc8765>.

[RFC8768] Boucadair, M., Reddy.K, T., and J. Shallow, "Constrained Application Protocol (CoAP) Hop-Limit Option", RFC 8768, DOI 10.17487/RFC8768, March 2020, <https://www.rfc-editor.org/info/rfc8768>.

[RFC8949] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", STD 94, RFC 8949, DOI 10.17487/RFC8949, December 2020, <https://www.rfc-editor.org/info/rfc8949>.

[RFC9147] Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", RFC 9147, DOI 10.17487/RFC9147, April 2022, <https://www.rfc-editor.org/info/rfc9147>.

[RFC9460] Schwartz, B., Bishop, M., and E. Nygren, "Service Binding and Parameter Specification via the DNS (SVCB and HTTPS Resource Records)", RFC 9460, DOI 10.17487/RFC9460, November 2023, <https://www.rfc-editor.org/info/rfc9460>.

[RFC9461] Schwartz, B., "Service Binding Mapping for DNS Servers", RFC 9461, DOI 10.17487/RFC9461, November 2023, <https://www.rfc-editor.org/info/rfc9461>.

[RFC9462] Pauly, T., Kinnear, E., Wood, C. A., McManus, P., and T. Jensen, "Discovery of Designated Resolvers", RFC 9462, DOI 10.17487/RFC9462, November 2023, <https://www.rfc-editor.org/info/rfc9462>.

[RFC9463] Boucadair, M., Ed., Reddy.K, T., Ed., Wing, D., Cook, N., and T. Jensen, "DHCP and Router Advertisement Options for the Discovery of Network-designated Resolvers (DNR)", RFC 9463, DOI 10.17487/RFC9463, November 2023, <https://www.rfc-editor.org/info/rfc9463>.

[STD13] Internet Standard 13, <https://www.rfc-editor.org/info/std13>.
At the time of writing, this STD comprises the following:

Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <https://www.rfc-editor.org/info/rfc1034>.

Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

10.2. Informative References

[BCP219] Best Current Practice 219, <https://www.rfc-editor.org/info/bcp219>. At the time of writing, this BCP comprises the following:

Hoffman, P. and K. Fujiwara, "DNS Terminology", BCP 219, RFC 9499, DOI 10.17487/RFC9499, March 2024, <https://www.rfc-editor.org/info/rfc9499>.

[BCP237] Best Current Practice 237, <https://www.rfc-editor.org/info/bcp237>. At the time of writing, this BCP comprises the following:

Hoffman, P., "DNS Security Extensions (DNSSEC)", BCP 237, RFC 9364, DOI 10.17487/RFC9364, February 2023, <https://www.rfc-editor.org/info/rfc9364>.

[CACHABLE-OSCORE] Amsüss, C. and M. Tiloca, "Cacheable OSCORE", Work in Progress, Internet-Draft, draft-amsuess-core-cachable-oscore-11, 6 July 2025, <https://datatracker.ietf.org/doc/html/draft-amsuess-core-cachable-oscore-11>.

[CoAP-CORR-CLAR] Bormann, C., "Constrained Application Protocol (CoAP): Corrections and Clarifications", Work in Progress, Internet-Draft, draft-ietf-core-corr-clar-03, 22 December 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-core-corr-clar-03>.

[CONSTR-NODES-BIS] Bormann, C., Ersue, M., Keränen, A., and C. Gomez, "Terminology for Constrained-Node Networks", Work in Progress, Internet-Draft, draft-ietf-iotops-7228bis-04, 2 March 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-iotops-7228bis-04>.

[CRI] Bormann, C. and H. Birkholz, "Constrained Resource Identifiers", Work in Progress, Internet-Draft, draft-ietf-core-href-30, 21 November 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-core-href-30>.

[DoC-paper] Lenders, M., Amsüss, C., Gündogan, C., Nawrocki, M., Schmidt, T., and M. Wählisch, "Securing Name Resolution in the IoT: DNS over CoAP", Proceedings of the ACM on Networking, vol. 1, no. CoNEXT2, pp. 1-25, DOI 10.1145/3609423, September 2023, <https://doi.org/10.1145/3609423>.

[REST] Fielding, R., "Architectural Styles and the Design of Network-based Software Architectures", Ph.D. Dissertation, University of California, Irvine, 2000, <https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf>.

[RFC3833] Atkins, D. and R. Austein, "Threat Analysis of the Domain Name System (DNS)", RFC 3833, DOI 10.17487/RFC3833, August 2004, <https://www.rfc-editor.org/info/rfc3833>.

[RFC6690] Shelby, Z., "Constrained RESTful Environments (CoRE) Link Format", RFC 6690, DOI 10.17487/RFC6690, August 2012, <https://www.rfc-editor.org/info/rfc6690>.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <https://www.rfc-editor.org/info/rfc7228>.

[RFC7828] Wouters, P., Abley, J., Dickinson, S., and R. Bellis, "The edns-tcp-keepalive EDNS0 Option", RFC 7828, DOI 10.17487/RFC7828, April 2016, <https://www.rfc-editor.org/info/rfc7828>.

[RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <https://www.rfc-editor.org/info/rfc7858>.

[RFC8094] Reddy, T., Wing, D., and P. Patil, "DNS over Datagram Transport Layer Security (DTLS)", RFC 8094, DOI 10.17487/RFC8094, February 2017, <https://www.rfc-editor.org/info/rfc8094>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC9052] Schaad, J., "CBOR Object Signing and Encryption (COSE): Structures and Process", STD 96, RFC 9052, DOI 10.17487/RFC9052, August 2022, <https://www.rfc-editor.org/info/rfc9052>.

[RFC9076] Wicinski, T., Ed., "DNS Privacy Considerations", RFC 9076, DOI 10.17487/RFC9076, July 2021, <https://www.rfc-editor.org/info/rfc9076>.

[RFC9176] Amsüss, C., Ed., Shelby, Z., Koster, M., Bormann, C., and P. van der Stok, "Constrained RESTful Environments (CoRE) Resource Directory", RFC 9176, DOI 10.17487/RFC9176, April 2022, <https://www.rfc-editor.org/info/rfc9176>.

[RFC9250] Huitema, C., Dickinson, S., and A. Mankin, "DNS over Dedicated QUIC Connections", RFC 9250, DOI 10.17487/RFC9250, May 2022, <https://www.rfc-editor.org/info/rfc9250>.

[RFC9528] Selander, G., Preuß Mattsson, J., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", RFC 9528, DOI 10.17487/RFC9528, March 2024, <https://www.rfc-editor.org/info/rfc9528>.

[TRANSPORT-IND] Amsüss, C. and M. S. Lenders, "CoAP Transport Indication", Work in Progress, Internet-Draft, draft-ietf-core-transport-indication-09, 7 July 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-core-transport-indication-09>.

Appendix A. Evaluation

The authors of this document presented the design, implementation, and analysis of DoC in their paper "Securing Name Resolution in the IoT: DNS over CoAP" [DoC-paper].

Acknowledgments

The authors of this document want to thank Mike Bishop, Carsten Bormann, Mohamed Boucadair, Deb Cooley, Vladimír Čunát, Roman Danyliw, Elwyn B. Davies, Esko Dijk, Gorry Fairhurst, Thomas Fossati, Mikolai Gütschow, Todd Herr, Tommy Pauly, Jan Romann, Ben Schwartz, Orie Steele, Marco Tiloca, Éric Vyncke, Tim Wicinski, and Paul Wouters for their feedback and comments.

Authors' Addresses

Martine Sophie Lenders
TUD Dresden University of Technology
Helmholtzstr. 10
D-01069 Dresden
Germany
Email: martine.lenders@tu-dresden.de

Christian Amsüss
Email: christian@amsuess.com

Cenk Gündoğan
NeuralAgent GmbH
Mies-van-der-Rohe-Straße 6
D-80807 Munich
Germany
Email: cenk.gundogan@neuralagent.ai

Thomas C. Schmidt
HAW Hamburg
Berliner Tor 7
D-20099 Hamburg
Germany
Email: t.schmidt@haw-hamburg.de

Matthias Wählisch
TUD Dresden University of Technology & Barkhausen Institut
Helmholtzstr. 10
D-01069 Dresden
Germany
Email: m.waehlisch@tu-dresden.de