CoRE
Internet-Draft
Intended status: Standards Track
Expires: 8 January 2026

C. Amsüss

M. S. Lenders
TU Dresden

7 July 2025

CoAP Transport Indication
draft-ietf-core-transport-indication-09

Abstract

The Constrained Application Protocol (CoAP, [RFC7252]) is available over different transports (UDP, DTLS, TCP, TLS, WebSockets), but lacks a way to unify these addresses. This document provides terminology and provisions based on Web Linking [RFC8288] and Service Bindings (SVCB, [RFC9460]) to express alternative transports available to a device, and to optimize exchanges using these.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://core-wg.github.io/transport-indication/. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-core-transport-indication/.

Discussion of this document takes place on the core Working Group mailing list (mailto:core@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/core/. Subscribe at https://www.ietf.org/mailman/listinfo/core/.

Source for this draft and an issue tracker can be found at https://github.com/core-wg/transport-indication.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 8 January 2026.

Copyright Notice

Copyright (c) 2025 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction

The Constrained Application Protocol (CoAP) provides multiple transports mechanisms: UDP and DTLS since [RFC7252], and TCP, TLS and WebSockets since [RFC8323]. Some additional transports being used in LwM2M [lwm2m], and even more being explored ([I-D.bormann-t2trg-slipmux], [I-D.amsuess-core-coap-over-gatt]). These are mutually incompatible on the wire, but CoAP implementations commonly support several of them, and proxies can translate between them.

CoAP currently lacks a way to indicate which transports are available for a given resource, and which endpoints are available for them. This document introduces ways to discover and how to use them.

CoAP also lacks a unified scheme to label a resource in a transport-independent way. This document does _not_ attempt to introduce any new scheme here, or raise a scheme to be the canonical one. Instead, each host or application can pick a canonical address for its resources, and advertise other transports in addition.

1.1. Terminology

Readers are expected to be familiar with the terms and concepts described in CoAP [RFC7252]. For web link based discovery, terminology introduced for link format [RFC6690] is used (or, equivalently, web links as described in [RFC8288]); for DNS based discovery, [RFC9460] provides the relevant definitions.

The phrase "the transport indicated by (a URI)" is used as described in Section 1.5.1.

A protocol that implements CoAP request-response semantics for a lower layer is called a "(CoAP) transport".

When the term "endpoint" is used in this document, it is generalized from the [RFC7252] definition to mean the transport and any multiplexing information particular to that transport.

1.2. Goals

This document introduces provisions for the seamless use of different transport mechanisms for CoAP. Combined, these provide:

1. Enablement: Inform clients of the availability of other transports of servers.

2. No Aliasing: Any URI aliasing must be opt-in by the server. Any defined mechanisms must allow applications to keep working on the canonical URIs given by the server.

3. Optimization: Do not incur per-request overhead from switching transports. This may depend on the server's willingness to create aliased URIs.

4. Proxy usability: All information provided must be usable by aware proxies to reduce the need for duplicate cache entries.

5. Proxy announcement: Allow third parties to announce that they provide alternative transports to a host.

For all these functions, security policies must be described that allow the client to use them as securely as the original transport.

This document will not concern itself with changes in transport availability over time, neither in causing them ("Please take up your TCP interface, I'm going to send a firmware update") nor in advertising their availability in advance. Hosts whose transport's availability changes over time can utilize any suitable mechanism to keep client updated, such as placing a suitable Max-Age value on their resources or having them observable.

1.3. Core principle: Transport endpoints are proxies

CoAP does not need any special provisions to send the same request for a single resource through different transports: A request to any globally addressable resource can be sent to any endpoint by phrasing it as a proxy request.

Whether that endpoint is trusted to, capable of and willing to relay that request, and how to find suitable endpoints to serve as a proxy for a request is discussed in this document.

When resource identifiers have different meanings depending on the host. the applicability of this document is limited.
// Possibly not limited a lot, but we have not looked into those
// cases in detail yet. --CA Examples of such resources are those whose URIs including loopback addresses or partially-qualified domain names.

1.4. Registered names and transport setup

The CoAP specification is intentionally open about the resolution services by which indirect identifiers in the host portion of a CoAP URI are resolved, giving DNS as one example without going into details (Section 6.1 of [RFC7252]).

This document does not change any of that, but it does point out in Section 2 the breadth of information that such a service can provide (e.g. providing multiple addresses, or metadata on services used on them), and gives a concrete example of the information that modern DNS idioms deliver (which it enables by performing a registration in Section 10.5).

1.5. Concepts

Same-host proxy: A CoAP server that accepts forward proxy requests (i.e., requests carrying the Proxy-Scheme option) exclusively for URIs that it is also the authoritative server for is defined as a "same-host proxy".

The distinction between a same-host and any other proxy is only relevant on a practical, server-implementation and illustrative level; this specification does not use the distinction in normative requirements, and clients need not make the distinction at all.

When talking of proxy requests, this document only talks of the Proxy-Scheme option. Given that all URIs this is usable with can be expressed in decomposed CoAP URIs, the need for using the Proxy-URI option should never arise. The Proxy-URI option is still equivalent to the decomposed options, and can be used if the server supports it.

1.5.1. Using URIs to identify transport endpoints

The URI coap://[2001:db8::1] identifies a particular resource, possibly a "welcome" text. It is, colloquially, also used to identify the combination of a CoAP transport and the transport specific details.

For precision, this document uses the term "the transport address indicated by (a URI)" to refer to the transport and its details, but otherwise no big deal is made of it.

Note that as such a URI may contain a registered name: as with any CoAP URI, resolution services apply. Therefore, it may indicate multiple transport endpoints.

[TBD: Do we want to extend this to HTTP proxies? Probably just not, and if so, only to those that can just take coap://... for a URI.]

2. Finding suitable endpoints for a URI

When a CoAP request is created by a client, a typical starting point is the URI of the request's target resource. To send the request, a suitable endpoint needs to be discovered. This section lists the ways one or more such endpoints can be found.

In some situations, a client decides to use a forward proxy to access the resource: either because it is explicitly configured to do so (Section 6), or because it has discovered a preferred proxy (Section 2.2). In that case, it finds the endpoint of the configured proxy (using Section 2.1, if not given explicitly). The proxy then decides on an endpoint to which to forward the request for its own, (again using the tools described in this section). Otherwise, it uses the information of scheme and authority, often through a resolution service (Section 2.1).

No matter how the endpoint (and thus transport) was discovered and whether a proxy is involved, it does not alter the URI of the resource being requested. The Proxy-Scheme, Uri-Host and Uri-Port options are set as needed to ensure that the request keeps targeting the requested resource. This happens, respectively, if the URI scheme associated with the selected transport differs from the request URI's scheme (or a proxy is used), when the host name is not the default one for the transport (e.g. if it is not an IP literal in the UDP or TCP cases, or a proxy is used; DNS CNAME entries or SVCB target do not alter the URI's host name at all) or a different port is used (as possible through SVCB). (Outside proxy cases, Section 6.4 of [RFC7252] only talks of setting the Uri-Host to preserve the URI, and not of setting Proxy-Scheme or Uri-Port. That is because at the time of writing, no mechanisms were available to select a different transport or port).

Note that there is no meaningful difference in a client's behavior between when it is configured with a proxy, has discovered a proxy through links or has discovered a completely different transport: this is the essence of "transport endpoints are proxies" Section 1.3.

// The following two paragraphs may need a new home eventually to
// keep this section to the point. --CA

For servers that follow the common pattern of exposing the same resources on all transports (and thus having multiple aliased URIs for the same resource) and that do not act as proxies for other

systems, the presence of the Proxy-Scheme option introduced by using alternative transports has little practical consequence: such servers become same-host proxies, and can ignore the Proxy-Scheme option as long as they recognize the Uri-Host value (which they already have been required to process).

While a server is at liberty to create aliases, clients can not infer from the presence of a transport for a host that URIs created from addressing that transport are present. For example, if coap://h.example.net/sensors/temp is a known resource, and CoAP-over-TCP on [2001:db8::1] is indicated as a transport endpoint, there is no reason for the client to assume that coap+tcp://[2001:db8::1]/sensors/temp exists, let alone is the same resource: Clients that access the known resource by establishing a TCP connection need to send the options Proxy-Scheme value "coap", the Uri-Host value "h.example.net" and the Uri-Path values "sensors" and "temp".

2.1. Processing scheme and authority

To discover endpoints for a given URI, the scheme and the authority component of the URI are typical starting points for resolution services.

The outcome of any resolution service is a list of transport candidates. It may be partially sorted, and may arrive piece by piece.

In addition to the list of transport candidates, a resolution service may also provide general metadata of the endpoint (e.g. necessary or sufficient requirements on the endpoint’s credentials, such as they are present in TLSA records).

Conceptually, each entry contains:

* Which CoAP transport is used (e.g. CoAP-over-TCP; typically expressed as an ALPN).

* The transport’s address details (e.g. an IP address and port number).

* Any additional metadata that facilitates communication (e.g. public keys for [I-D.ietf-tls-esni]).

While it is generally preferable for the lookup result to be complete, it might manage only partial information. From that partial information, further resolution may be performed, or the information may already suffice to send a request based on other known proxy information.

Resolution services may have more structure in that list, may provide multiple choices in a single result or may branch out in multiple layers. At a conceptual level, those are reduced to an expanded list of individual candidates.

2.1.1. Transport-unaware resolution

The IP based transports specified so far (CoAP over UDP, DTLS, TCP, TLS and WebSockets) all indicate the transport in their scheme, and either use their default port or indicate the port explicitly. The only remaining details of multiplexing information required are the IP version(s) and IP address(es) of the server.

If the host component of the URI is a literal, that information is already available.

If the host component of the URI is a registered name, a name resolution service is used for a simple name lookup: When DNS is used as a resolution service, AAAA (or A) records of the name are looked up. The /etc/hosts file and nsswitch "host" database can provide that same information.

Additional metadata provided by this resolution service are the zone identifier (implied in DNS by using the zone the DNS response was obtained through) or additional records (such as khe aforementionened TLSA records).

Simple resolution services do not indicate which transports are available on the address. Servers reached that way can resort to Section 2.2 to indicate alternative transports while exchanging initial data through the original transport, or to store information in link format / web-link based information systems (such as a Resource Directory [RFC9176]).

2.1.2. Transport-aware resolution mechanisms

Advanced resolution services provide information about which transports are available in addition to those of the previous section.

For the DNS resolution mechanism, SVCB lookups described in Section 7.1 provide that information.

It is recommended that future transports are designed to utilize transport-aware resolution mechanisms; see Section 8 for details.

2.2. Explicit proxy indication

A server can advertise a recommended proxy by publishing a Web Link with the "has-proxy" relation, defined in this document, to a URI indicating its transport address. In particular (and that is a typical case), it can indicate its own network address on an alternative transport when implementing same-host proxy functionality.

The semantics of a link from S to P with relations has-proxy ("S has-proxy P", <P>;rel=has-proxy;anchor="S") are that for any resource that has the same origin as S, the transport address indicated by P can be used to obtain that resource.

2.2.1. Example

A constrained device at the address 2001:db8::1 that supports CoAP over TCP in addition to CoAP can self-describe like this:

```
Req: to [ff02::fd]:5683 on UDP
Code: GET
Uri-Path: /.well-known/core
Uri-Query: if=tag:example.com,sensor

Res: from [2001:db8::1]:5683
Content-Format: application/link-format
Payload:
</sensors/temp>;if="tag:example.com,sensor",
<coap+tcp://[2001:db8::1]>;rel=has-proxy;anchor="/"


Req: to [2001:db8::1]:5683 on TCP
Code: GET
Proxy-Scheme: coap
Uri-Path: /sensors/temp
Observe: 0

Res: 2.05 Content
Observe: 0
Payload:
39.1属C
```

Figure 1: Discovery and follow-up request through a has-proxy relation

The discovery process yields two links: The first describes the resource, the second describes that an additional (TCP) endpoint is available for all resources on this host.

Note that generating this discovery file needs to be dynamic based on its available addresses; only if queried using a link-local source address, the server may also respond with a link-local address in the authority component of the proxy URI.

3. Operational concerns of discovered transport endpoints

3.1. Security context propagation

Any security requirements posed by a server or client application on a CoAP request MUST be applied independently of the transport that is used to perform the request. If a transport can not be used to satisfy the requirements, it is ineligible for use with the request (from a client's point of view), and unauthorized (from a server's point of view).

If the requirements contain transport layer security, the proxy needs to present the credentials required of the server to the client, and those of the client to the server; this is only practical when the proxy is a same-host proxy.

Some applications have requirements exceeding the requirements of a secure connection, e.g., (explicitly or implicitly) requiring that name resolution happen through a secure process and packets are only routed into networks where it trusts that they will not be intercepted on the path to the server. Such applications need to extend their requirements to the the sources used to obtain the endpoints (i.e., the source of any has-proxy statement or the SVCB data); a sufficient (but maybe needlessly strict) requirement for has-proxy statements is to only follow those that are part of the same resource that advertises the link currently being followed. Section Section 9.2 adds further considerations.

3.1.1. Exception: Narrowing security requirements

If a concrete application starts with a minimal set of security requirements, has no means of discovering security properties of the endpoint and can only discovery security properties of a transport, it MAY describe how a first step of transport discovery narrows the security requirements.

An example of this is Section 3.2 of [I-D.ietf-core-dns-over-coap] where the target name of the SVCB record is used to set the hostname component of a URI and thus set security requirements.

Before making use of this option, alternatives such as using TLSA records should be exhausted.

3.2. Choice of endpoints

It is up to the client whether to use an advertised endpoint, or (if multiple are provided) which to pick.

Information about endpoints may be annotated with additional metadata that may help guide such a choice; defining such metadata is out of scope for this document.

Clients MAY switch between endpoints as long as the source describing them is fresh; they may even do so per request. (For example, they may perform individual requests using CoAP-over-UDP, but choose CoAP-over-TCP for requests with large expected responses). When the information about endpoints is obtained through CoAP (e.g. as a has-proxy link), the client can use the describing representation's ETag to efficiently renew its justification for using the alternative transport.

3.3. Selection of a canonical origin

While a server is at liberty to provide the same resource independently on different transports (i.e. to create aliases), it may make sense for it to pick a single scheme and authority under which it announces its resources. Using only one address helps proxies keep their caches efficient, and makes it easier for clients to avoid exploring the same server twice from different angles.

When there is a predominant scheme and authority through which an existing service is discovered, it makes sense to use these for the canonical addresses.

Otherwise, it is suggested to use the coap or coaps scheme (given that these are the most basic and widespread ones), and the most stable usable name the host has.

3.3.1. Unreachable canonical origin addresses

For devices that are not generally reachable at a stable address, it may make sense to use a scheme and authority as the canonical address that can not actually be dereferenced.

The registered names available for that purpose depend on the resolution mechanisms in use. When the Domain Name System (DNS) is used, such names would not be associated with any A or AAAA records (but may still use, for example, TLSA records).

Such URIs are _only_ usable to clients that discover a suitable proxy along with the URI, and which can place sufficient trust in that proxy.

3.4. Advertisement through a Resource Directory

In the Resource Directory specification [rfc9176], protocol negotiation was anticipated to use multiple base values. This approach was abandoned since then, as it would incur heavy URI aliasing.

Instead, devices can submit their has-proxy links to the Resource Directory like all their other metadata.

A client performing resource lookup can ask the RD to provide available (same-host-)proxies in a follow-up request by asking for ?anchor=<the-discovered-host>&rel=has-proxy. The RD may also volunteer that information during resource lookups even though the has-proxy link itself does not match the search criteria.

[

It may be useful to define RD parameters for use with lookup here, which'd guide which available proxies to include. For example, asking ?if=tag:example.com,sensor&proxy-links=tcp could give as a result:

<coap://[2001:db8::1]/s>;rt=tag:example.com,sensor,<coap+tcp://[2001:db8::1]/>;rel=has-proxy;anchor="coap://[2001:db8::1]/"

This is similar to the extension suggested in Section 5 of [I-D.amsuess-core-resource-directory-extensions].

]

4. Elision of Proxy-Scheme and Uri-Host

A CoAP server may publish and accept multiple URIs for the same resource, for example when it accepts requests on different IP addresses that do not carry a Uri-Host option, or when it accepts requests both with and without the Uri-Host option carrying a registered name. Likewise, the server may serve the same resources on different transports. This makes for efficient requests (with no Proxy-Scheme or Uri-Host option), but in general is discouraged [aliases].

To make efficient requests possible without creating URI aliases that propagate, the "has-unique-proxy" specialization of the has-proxy relation and the "is-unique-proxy" SVCB parameter are defined.

If a proxy is unique, it means that requests arriving at the proxy are treated the same no matter whether the scheme, authority and port of the link context are set in the Proxy-Scheme, Uri-Host and Uri-Port options, respectively, or whether all of them are absent.

[The following two paragraphs are both true but follow different approaches to explaining the observable and implementable behavior; it may later be decided to focus on one or the other in this document.]

While this creates URI aliasing in the requests as they are sent over the network, applications that discover a proxy this way should not "think" in terms of these URIs, but retain the originally discovered URIs (which, because Cool URIs Don’t Change[cooluris], should be long-term usable). They use the proxy for as long as they have fresh knowledge of the has-(unique-)proxy statement.

In a way, advertising has-unique-proxy can be viewed as a description of the link target in terms of SCHC [I-D.ietf-lpwan-coap-static-context-hc]: In requests to that target, the link source’s scheme and host are implicitly present.

While applications retain knowledge of the originally requested URI (even if it is not expressed in full on the wire), the original URI is not accessible to caches both within the host and on the network (for the latter, see Section 6). Thus, cached responses to the canonical and any aliased URI are mutually interchangeable as long as both the response and the proxy statement are fresh.

A client MAY use a unique-proxy like a proxy and still send the Proxy-Scheme and Uri-Host option; such a client needs to recognize both relation types, as relations of the has-unique-proxy type are a specialization of has-proxy and typically don’t carry the latter (redundant) annotation. [To be evaluated -- on one hand, supporting it this way means that the server needs to identify all of its addresses and reject others. Then again, is a server that (like many now do) fully ignore any set Uri-Host correct at all?]

Example:

```
Req: to [ff02::fd]:5683 on UDP
Code: GET
Uri-Path: /.well-known/core
Uri-Query: if=tag:example.com,sensor

Res: from [2001:db8::1]:5683
Content-Format: application/link-format
Payload:
</sensors/>;if="tag:example.com,collection",
<coaps+ws://[2001:db8::1]>;rel=has-unique-proxy;anchor="/"


Req: to [2001:db8::1] via WebSockets over HTTPS
Code: GET
Uri-Path: /sensors/

Res: 2.05 Content
Content-Format: application/link-format
Payload:
</sensors/temperature>;if="tag:example.com,sensor"
```

Figure 2: Follow-up request through a has-unique-proxy relation. Compared to the last example, 5 bytes of scheme indication are saved during the follow-up request.

It is noteworthy that when the URI reference /sensors/temperature is resolved, the base URI is coap://device0815.example.com and not its coaps+ws counterpart -- as the request is still for that URI, which both the client and the server are aware of. However, this detail is of little practical importance: A simplistic client that uses coaps+ws://device0815.proxy.rd.example.com as a base URI will still arrive at an identical follow-up request with no ill effect, as long as it only uses the wrongly assembled URI for dereferencing resources, the security context is the same, the state is kept no longer than the has-unique-proxy statement is fresh, and it does not (for example) pass the URI on to other devices.

4.1. Impact on caches

[This section is written with the "there is implied URI aliasing" mindset; it should be possible to write it with the "compression" mindset as well (but there is no point in having both around in the document at this time).

It is also slightly duplicating, but also more detailed than, the brief note on the topic in Section 6]

When a node that performs caching learns of a has-unique-proxy statement, it can utilize the information about the implied URI aliasing: As long as the has-unique-proxy statement is fresh and trusted, requests for either of the origins can be served from the cache of the other origin.

4.2. Using unique proxies securely

The elision of the host name afforded by the unique-proxy relation is only possible if the required security mechanisms verify the scheme and host of the server.

This is given for OSCORE based mechanisms, where "unprotected message fields (including Uri-Host [...]) MUST not lead to an OSCORE message becoming verified".

With TLS based security mechanisms, name and scheme can not be completely elided in general. While the use of the SNI HostName field sets the default Uri-Host already, the scheme still needs to be sent in a Proxy-Scheme option to satisfy the requirement of Section 3.1.

[It may be possible to relax this requirement if the host publishes a _trustworthy_ statement about serving the same content on all schemes; however, no urgent need for this optimization is currently known that warrants the extra scrutiny.]

4.3. Self-description as a unique proxy

The level of assurance a client needs from a server to elide the Uri-Host option in a request that was created from a URI with no IP address literal has been a controversial topic. [Should we dig up old conversations, link to https://github.com/core-wg/wiki/wiki/CoAP-FAQ#q4, or just let the weight of a WG consensus-document-to-be do its work?]

The has-unique-proxy relation provides an easy way for a server to indicate that this is in fact allowed: A server can publish a statement such as </>;rel=has-unique-proxy in its /.well-known/core file. A client that receives and understands it can thus elide the Uri-Host option in requests to the server as per the definition of the relation.

5. Third party proxy services

A server that is aware of a suitable cross proxy may use the has-proxy relation to advertise that proxy. If the transport used towards the proxy provides name indication (as CoAP over TLS or WebSockets does), or by using a large number of addresses or ports, it can even advertise a (more efficient) has-unique-proxy relation. This is particularly interesting when the advertisements are made available across transports, for example in a Resource Directory.

How the server can discover and trust such a proxy is out of scope for this document, but generally involves the same kind of links. In particular, a server may obtain a link to a third party proxy from an administrator as part of its configuration.

The proxy may advertise itself without the origin server's involvement; in that case, the client needs to take additional care (see Section 9.2).

```
Req: GET http://rd.example.com/rd-lookup?if=tag:example.com,sensor

Res:
Content-Format: application/link-format
Payload:
<coap://device0815.example.com/sensors/>;if="tag:example.com,collection",
<coap+wss://device0815.proxy.rd.example.com>;rel=has-unique-proxy;anchor="coap://device08
15.example.com/"


Req: to device0815.proxy.rd.example.com on WebSocket
Host (indicated during upgrade): device0815.proxy.rd.example.com
Code: GET
Uri-Path: /sensors/

Res: 2.05 Content
Content-Format: application/link-format
Payload:
</sensors/temperature>;if="tag:example.com,sensor"
```

Figure 3: HTTP based discovery and CoAP-over-WS request to a CoAP resource through a has-unique-proxy relation

5.1. Generic proxy advertisements

A third party proxy may advertise its availability to act as a proxy for arbitrary CoAP requests. This use is not directly related to the transport indication in other parts of this document, but sufficiently similar to warrant being described in the same document.

The resource type "CPA-core.proxy" can be used to describe such a proxy.

```
Req: GET coap://[fe80::1]/.well-known/core?rt=CPA-core.proxy

Res:
Content-Format: application/link-format
Payload:
<>;rt=CPA-core.proxy

Req: to [fe80::1] via CoAP
Code: GET
Proxy-Scheme: http
Uri-Host: example.com
Uri-Path: /motd
Accept: text/plain

Res: 2.05 Content
Content-Format: text/plain
Payload:
On Monday, October 25th 2021, there is no special message of the day.
```

Figure 4: A CoAP client discovers that its border router can also serve as a proxy, and uses that to access a resource on an HTTP server.

The considerations of Section 9.2 apply here.

A generic advertised proxy is always a forward proxy, and can not be advertised as a "unique" proxy as it would lack information about where to forward.

A proxy may be limited in the URIs it can service, for technical reasons (e.g. when none of the URI’s transports are supported by the server) or for policy reasons (only accessing servers inside an organizational structure). Future documents (or versions of this document) may add target attributes that allow specifying the capabilities of a proxy. [An earlier version of this document contained a proxy-schemes attribute. This was discontinued because it could already not express whether a proxy could access IPv4 or IPv6 peers, and because the use of schemes is becoming less useful given the new recommendation of incorporating details from registered name resolution into the transport selection.]

The use of a generic proxy can be limited to a set of devices that have permission to use it. Clients can be allowed by their network address if they can be verified, or by using explicit client authentication using the methods of [I-D.tiloca-core-oscore-capable-proxies].

6. Client picked proxies

This section is purely informative, and serves to illustrate that the mechanisms introduced in this document do not hinder the continued use of existing proxies.

When a resource is accessed through an "actual" proxy (i.e., a host between the client and the server, which itself may have a same-host proxy in addition to that), the proxy's choice of the upstream server is originally (i.e., without the mechanisms of this document) either configured (as in a "chain" of proxies) or determined by the request URI (where a proxy picks CoAP over TCP and resolves the given name for a request aimed at a coap+tcp URI).

A proxy that has learned, by active solicitation of the information or by consulting links in its cache, that the requested URI is available through a (possibly same-host) proxy, may use that information in choosing the upstream transport, to correct the URI associated with a cached response, and to use responses obtained through one transport to satisfy requests on another.

For example, if a host at coap://h1.example.com has advertised </res>,<coap+tcp://h1.example.com>;rel=has-proxy;anchor="/", then a proxy that has an active CoAP-over-TCP connection to h1.example.com can forward an incoming request for coap://h1.example.com/res through that CoAP-over-TCP connection with a suitable Proxy-Scheme and Uri-Host on that connection.

If the host had marked the proxy point as <coap+tcp://h1.example.com>;rel=has-unique-proxy instead, then the proxy could elide the Proxy-Scheme and Uri-Host options, and would (from the original CoAP caching rules) also be allowed to use any fresh cache representation of coap+tcp://h1.example.com/res to satisfy requests for coap://h1.example.com/res.

A client that uses a forward proxy and learns of a different proxy advertised to access a particular resource will not change its behavior if its original proxy is part of its configuration. If the forward proxy was only used out of necessity (e.g., to access a resource whose indicated transport not supported by the client) it can be practical for the client to use the advertised proxy instead.

7. Service Binding Parameters for CoAP transports

Discovery mechanisms that exist in DNS [RFC9460], DHCP, Router Advertisements [RFC9463] or other mechanisms can provide details already that would otherwise only be discovered later through proxy links. For when those details are provided in the shape of Service Binding Parameters, this section describes their interpretation in the context of CoAP transport indication.

This document is an SVCB mapping document as described in [RFC9460]. As a non-normative overview, the mapping summary (following the template of Appendix B of [RFC9460]) is as follows:

Mapped schemes	"coap", "coaps", "coap+tcp", "coaps+tcp", "coap+ws", "coaps+ws"
RR type	SVCB (64)
Name prefix	_coap for the scheme's default port, else _$PORT._coap
Automatically Mandatory Keys	port, no-default-alpn
SvcParam defaults	alpn: according to scheme (or empty)
Special behaviors	Multiple mapped schemes; alpn default influenced by initial URI.
Keys that records must include	None

Table 1

[The following paragraph is outdated, but its replacement will depend on the outcome of IETF122 discussions.]

The subsections in this section are arranged to describe a consistent sequential full picture. The capabilities of this big picture are not exercised by any application known at the time of draft publication. It is instead backed by many small-scope use cases (such as bootstrapping issues of proxy-link based CoAP scheme unification, [I-D.lenders-core-dnr], [I-D.amsuess-t2trg-onion-coap] and also applications outside CoAP such as [SUIB]) and presents a unified solution framework.

7.1. Discovering transport indication details from name resolution

This document registers the _coap attrleaf label in Section 10.5 using the pattern described as described in Section 10.4.5 of [RFC9460], and thus enables the use of SVCB records. This path is chosen over the creation of a new SVCB RR "COAP" because it is considered unlikely that DNS implementations would update their code bases to apply SVCB behavior; this assumption will be revisited before registration.

These can be used during the resolution of URIs that use any CoAP scheme. The presence of an SVCB record for a registered name implies that any transport advertised in the record is suitable for proxying to resources of any CoAP scheme and that registered name, provided that a resource is available at that URI in the first place. This does not create URI aliasing: Any resource is still accessed at its original URI through the advertised proxy endpoints.

It is possible through this to advertise transports without transport layer security for URIs with the schemes "coaps", "coaps+tcp" and "coaps+ws". Unless the application explicitly regards an object layer security mechanism as a sufficient replacement for transport layer security, those transports can not be selected for operations on such URIs as per Section 3.1.

Some SVCB parameters have defaults; for "_coap", these are:

* port: 5683

* ALPN: empty

As SVCB records were not specified for the existing CoAP transports originally, generic CoAP clients are not required to use the SVCB lookup mechanism, but MAY attempt it opportunistically in order to obtain a usable transport (or to obtain it faster). Applications built on CoAP MAY require clients to perform this kind of discovery. Adding such a requirement is particularly useful if the application frequently advertises URIs with a scheme that defaults to a transport which its clients may not support, or when the application makes use

of functionality afforded by [RFC9460] such as apex domain redirection. (Had the SVCB specification predated the first new CoAP transports, that mechanism might have been used in the first place instead of additional schemes).

[The following paragraph may need to be revisited depending on the outcome of IETF122 discussions.]

The effects on a client of seeing SVCB parameters are similar to those of seeing a "has-proxy" link from the origin to the URI built using {#svcblit}. The precise implications of any DNSSec-backed applicable credentials expressed in SVCB parameters are subject to ongoing discussion (especially with regard to whether they apply to the proxy or the origin server).

7.2. Service Parameters

Several parameters are relevant in the context of CoAP, independently of whether they are used with SVCB records or Service Binding Parameters transported outside SVCB records, and independently of whether they apply to the _coap service or another service that can be used on top of CoAP (such as _dns):

* port: The CoAP service using the transport described in this parameter is reachable on this port (described in [RFC9460]). It is automatically mandatory.

  This document does not limit which ports can be used with CoAP.

* alpn: The ALPN "coap" has been defined for CoAP-over-TLS [RFC8323], and "co" for CoAP-over-DTLS in [I-D.ietf-core-coap-dtls-alpn].

  If an ALPN service parameter is found, this indicates that the ALPN(s) and thus the CoAP transport that can be used on this address / port. For example, "co" indicates that DTLS (and thus UDP) is used.

  ALPN IDs are defined in this document and in [I-D.ietf-core-coap-dtls-alpn] for the CoAP transports that had no such identifier when they were specified.

  The default value of alpn when processing a URI (as in Section 2.1) is the ALPN corresponding to the used scheme. The default value when not processing a URI (e.g. when processing service parameters from DNR [RFC9463]) is empty.

* no-default-alpn, mandatory, ipv4hint, ipv6hint: Used as described in Section 7 of [RFC9460].

* is-unique-proxy: This is a new parameter defined in this document, and equivalent to the has-unique-proxy in its semantics.

  Its value is empty.

* Applications may extend the supported parameters. In particular, [I-D.ietf-core-dns-over-coap] has already introduced the docpath parameter which indicates a path at which DNS-over-CoAP is available at the given address.

The following parameters are under consideration for inclusion in this list, but it is unsure whether they are suitable. Those parameters would describe the origin server and not the individual (proxy) transport through which it can be reached.

* cred: This is a new parameter defined in this document, and describes COSE credentials that can authenticate the server, e.g. when used with EDHOC.

  The cred parameter’s value is a CBOR sequence of COSE Header maps as defined in [RFC9052]. If the parameter is present, it indicates that the server can be authenticated by any credential expressed in the sequence. This is similar to the TLSA records specified in [RFC6698].

  A COSE Header map can express many properties, not all of which are sufficient to authenticate a peer on any given security mechanism. Without excluding applications that may process other entries, a practical criterion for whether a header map is suitable for EDHOC is that the header map could also be used in EDHOC as ID_CRED_R if the credential is sent by value.

  For example, a header map with a kccs entry can be used to indicate a public key including a Key ID (kid), and that public key does not need to be sent during the EDHOC exchange. Alternatively, a header map with an x5t identifies the end entity certificate the server presents by a thumbprint (hash).

  It is up to the application to define requirements for the provenance of the cred parameter, whether it needs to be provided through secure mechanism, or whether the server is strictly required to present that credential.

This is unlike TLSA, which needs to be transported through DNSSEC, because a cred parameter may be sent using other means than DNS (for example in DHCPv6 responses or Router Advertisements).

* edhoc-info: This is a new parameter defined in this document, describing how EDHOC can be used on the server.

  The value of the parameter is a CBOR array following the APP_PROF_SEQ structure defined in [I-D.tiloca-lake-app-profiles].

  It is optional to provide and optional to process, but can help speed up the establishment of a security context.

* oauth-hints: This is a new parameter defined in this document, describing how ACE-OAuth [RFC9200] can be used with this service.

  Its value is a CBOR map containing AS Request Creation Hints as described in Section 5.3 of [RFC9200]. While an empty map can be useful (hinting that the client should use its configured ACE-OAuth setup), typical useful keys are 1 (AS, indicating the URI of the Authorization Server), 5 (audience, indicating the name under which the service is known to the Authorization Server), and 9 (scope, when discovering a particular service rather than just getting transport information for a host). That data is using the same shape the server might use when responding to an attempt at an unencrypted connection, and can not only speed up the discovery of the right AS, but can also protect that information (e.g. when DNSSEC is used), and avoids the need for an unprotected first request.

  It is up to the application to define requirements for the use of such data. For example, it may require that the audience matches the requested host name, or may require that the scope matches the kind of service being discovered.

  When expressed in text format, e.g. in DNS zone files, the CBOR diagnostic notation [I-D.ietf-cbor-edn-literals] can be used.

7.2.1. Examples of using name resolution discovery and parameters

7.2.1.1. Generic client discovering transport options

A generic client is directed to obtain coap://dev1.example.com/log requests the name to be resolved using the system’s resolution mechanisms, resulting in a DNS query for these records:

```
_coap.dev1.example.com IN SVCB
dev1.example.com       IN AAAA
```

The following records are returned:

```
_coap.dev1.example.com IN SVCB 1 . alpn=COAP,CO
_coap.dev1.example.com IN SVCB 1 . alpn=co,coap port=5684
_coap.dev1.example.com IN SVCB 1 . alpn=CO port=61616
dev1.example.com       IN AAAA 2001:db8:1::1
```

Exceeding the single option it had with just the IP address, it may now also choose to establish a TCP connection on the default port, to use port 61616 for UDP (which results in more compact frames on a 6LoWPAN link), or to use either of the TLS based options.

7.2.1.2. Application mandating SVCB

An example application's policy is to mandate client support for SVCB records, and to require that a security mechanism must be used where credentials are backed either by DNSSEC or by the Web PKI.

A server operator is running in a legacy network that only provides an IPv4 address behind NAT with a dynamic public address, but has PCP [RFC7291] available. After running PCP to open a UDP port, it learns that 1.2.3.4:5678 will be available for some time.

It therefore updates its DNS record like this:

```
_coap.host.example.net 600 IN SVCB 1 publicudp.host.example.net       \
                       port=5678                                      \
                       cred={14:{... /KCCS with its public key/}}
```

When a client starts using coap://host.example.net/interactive, it looks up that record and verifies it using DNSSEC. It then proceeds to send EDHOC requests over CoAP to 1.2.3.4 port 5678, setting the Uri-Host option to "host.example.net".

The client could also have initiated an EDHOC session if no cred parameter had been present, but then, it would have required that the server present some credential that could be verified through the Web PKI, for example an x5chain containing a Let's Encrypt certificate.

7.3. Producing requests for a discovered service

If a service's discovery process does not produce a URI but an address, host name and/or Service Binding Parameters, those can be converted to a CoAP URI, for which transport hints are already encoded in the parameters the URI is constructed from. An example of this is DNS server discovery [I-D.ietf-core-coap-dtls-alpn].

While it is up to the service to define the service's semantics, this section applies to any service whose use with CoAP is defined by a normative referencing this section:

* The client tries the available services with their ALPNs and CoAP transports according to its capabilities and the priorities and mandatory parameters as defined for Service Bindings.

* The service either defines a well-known path, or it defines a Service Binding Parameter that describes the service's path on the described endpoint, or it defines both (and the well-known path is the default in absence of the defined parameter).

  The value is a CBOR sequence [RFC8742] of text strings, which represent Uri-Path options in a CoAP request, or (equivalently) the path of a CRI reference [I-D.ietf-core-href].

  A parameter value that is not a well-formed CBOR sequence, or any item is not a text string, is considerd malformed.

  When expressed in text format, e.g. in DNS zone files, the CBOR diagnostic notation [I-D.ietf-cbor-edn-literals] can be used.

  To access the service, a client sets the text string values of the used Service Binding parameter as Uri-Path options in the request.

  If the resource is unavailable, the client may continue with options that have a larger SvcPriority value associated (if such a property exists in the discovery method).

  An example of such an option is docpath as defined in [I-D.ietf-core-dns-over-coap]. (As that document precedes this one, it repeats the same rules explicitly rather than reusing these rules).

* A Service Binding is accompanied by a hostname: For example, this is the hostname of the Encrypted DNS Resolver or the Special-Use Domain Name in the case of [RFC9462] lookups, or the authentication-domain-name in case of [RFC9463] DHCP options or Router Advertisements.

  Unless its value is identical to the default value for Uri-Host (which is the case on transports with Server Name Indication (SNI)), the that name is added in the Uri-Host option.

* If the port Service Binding Parameter is set, the Uri-Port option is set to the port that set in the port prefix of the query (or the used CoAP transport's default port), unless that is its default value anyway.

* No Proxy-Scheme option is set.

By following the rules of Section 6.5 of [RFC7252] or the equivalent rules for the respective CoAP transport, the service can be translated into a URI. This implies URI aliasing between the composed URIs of all transports if any of the transports use different schemes.

The rules for setting Uri-Host and Uri-Port result in the authority component of the URI being equal to the Binding Authority defined in [RFC9461].

Note that since different security policies may apply to service discovery and other application components that dereference URIs, any connections established while using the service and cache entries created by it need to be treated carefully, for example by using separate connection and cache pools.

7.4. Expressing Service Parameters as literals

A method for expressing Service Parameters in URIs that do not use registered names is described in Appendix E.

Among other things, that mechanism allows encoding the full information obtained during service discovery in a URI instead of just the one choice taken. It is also required if different CoAP transports are using the same scheme (as is recommended in Section 8) with IP address literals in URIs, for which unlike for resolved names no service parameters are available.

8. Guidance to upcoming transports

When new transports are defined for CoAP, it is recommended to use the "coap" scheme (or "coaps" for TLS based transports).

If the transport's identifiers are IP based and have identifiers typically resolved through DNS, authors of new transports are encouraged to specify how they are expressed in Service Binding records ([RFC9460]), e.g., using an alpn parameter (which does not necessarily indicate the use of a TLS based transport), and if IP literals are relevant to the transport, to follow up on Appendix E.

If the transport's native identifiers are compatible with the structure of the authority component of a URI, those identifiers can be used as an authority as-is. To help the host decide the resolution mechanism, it may be helpful to register a subdomain of .arpa as described in [rfc3172]. The guidence for users is to never attempt to resolve such a name, and for the zone's implementation is to return NXDOMAIN unconditionally.

If the transport's native identifiers are incompatible with that structure (e.g. because they contain colons), the document may define some transformation.

If a transport's native identifiers are only local, the zone .alt [rfc9476] may be used instead.

For example, CoAP over GATT [I-D.amsuess-core-coap-over-gatt] removes the colons from Bluetooth Low Energy MAC addresses like 00:11:22:33:44:55 and combines them into authority components such as 001122334455.ble.arpa. Slipmux [I-D.bormann-t2trg-slipmux] might use the locally significant device name /dev/ttyUSB0 as coap://ttyUSB0.dev.alt/.

URIs created from such names may not indicate the protocol uniquely: Additional transports specified later may also provide CoAP services for the same name. In the sense of Section 1.5.1, both transport would be identified by that URI. That is not an issue as long as the protocols underneath the CoAP transport provide a means of advertising the precise protocol used. For example, a hypothetical CoAP transport for BLE that is not GATT based can be selected for the same scheme and authority based on data in the BLE advertisement.

9. Security considerations

9.1. Security context propagation

Clients need to strictly enforce the rules of Section 3.1. Failure to do so, in particular using a thusly announced proxy based on a certificate that attests the proxy's name, would allow attackers to circumvent the client's security expectation.

When security is terminated at proxies (as is in DTLS and TLS), a third party proxy can usually not satisfy this requirement; these transports are limited to same-host proxies.

9.2. Traffic misdirection

Accepting arbitrary proxies, even with security context propagation performed properly, would allow attackers to redirect traffic through systems under their control. Not only does that impact availability, it also allows an attacker to observe traffic patterns.

This affects both OSCORE and (D)TLS, as neither protect the participants' network addresses.

Other than the security context propagation rules, there are no hard and general rules about when an advertised proxy is a suitable candidate. Aspects for consideration are:

* When no direct connection is possible (e.g. because the resource to be accessed is served as coap+tcp and TCP is not implemented in the client, or because the resource's host is available on IPv6 while the client has no default IPv6 route), using a proxy is necessary if complete service disruption is to be avoided.

  While an adversary can cause such a situation (e.g. by manipulating routing or DNS entries), such an adversary is usually already in a position to observe traffic patterns.

* A proxy advertised by the device hosting the resource to be accessed is less risky to use than one advertised by a third party.

  The /.well-known/core resource is regarded as a source of authoritative information on the endpoint's CoAP related metadata, and can be queried early in the discovery process, or queried for verification (with filtering applied) after discovery through an RD. Other resources may be less trustworthy as they may be controlled by entities not trusted with the endpoint's traffic.

Appendix D describes an extension to [I-D.ietf-lake-edhoc] by which the client can verify that the proxy used by the client is recognized by the server. This is similar to querying /.well-known/core for any proxies advertised there, but happens earlier in the connection establishment, and leaves the decision whether the proxy is legitimate to the server.

It only conveys information about the URI of the proxy. The mapping of any host name inside it to an IP address, or of an IP address to a routing decision, is left to the security mechanisms of the respective layers.

9.3. Protecting the proxy

A widely published statement about a host's availability as a proxy can cause many clients to attempt to use it.

This is mitigated in well-behaved clients by observing the rate limits of [RFC7252], and by ceasing attempts to reach a proxy for the Max-Age of received errors.

Operators can further limit ill-effects by ensuring that their client systems do not needlessly use proxies advertised in an unsecured way, and by providing own proxies when their clients need them.

10. IANA considerations

10.1. Link Relation Types

IANA is asked to add two entries into the Link Relation Type Registry last updated in [RFC8288]:

Relation Name	Description	Reference
has-proxy	The link target can be used as a proxy to reach URIs inside the origin of the link context.	RFCthis
has-unique-proxy	Like has-proxy, and using this proxy implies scheme and host of the target.	RFCthis

Table 2: New Link Relation types

10.2. Resource Types

IANA is asked to add an entry into the "Resource Type (rt=) Link Target Attribute Values" registry under the Constrained RESTful Environments (CoRE) Parameters:

[The RFC Editor is asked to replace any occurrence of CPA-core.proxy with the actually registered attribute value.]

Attribute Value: core.proxy

Description: Forward proxying services

Reference: [this document]

10.3. TLS Application-Layer Protocol Negotiation (ALPN) Protocol IDs

IANA is requested to add the following entries to the TLS Application-Layer Protocol Negotiation (ALPN) Protocol IDs registry ([RFC7301]).

The identification sequences are suggestions to be approvded by the reviewers

Protocol	Identification sequence	Reference
CoAP (over UDP)	0x43 0x4f ("CO")	[RFC7252]
CoAP (over TCP)	0x43 0x4f 0x41 0x50 ("COAP")	[RFC8323]

Table 3

10.4. Service Parameter Key (SvcParamKey)

IANA is NOT YET requested to add the following entries to the SVCB Service Parameters registry ([RFC9460]). The definition of this parameter can be found in Section 8.

Number	Name	Meaning
to be assigned	cred	COSE credentials identifying the server
to be assigned	edhoc-info	EDHOC profile information
to be assigned	oauth-hints	Describes how to obtain a token at an ACE Authorization Server

Table 4

All entries have in common that their Reference is this this document, Section 7.2}, and that their change controller is IETF.

10.5. Underscored and Globally Scoped DNS Node Names

IANA is NOT YET requested to add the following entry to the Underscored and Globally Scoped DNS Node Names registry (in the DNS Parameters group) established in [RFC8552] and thus enables its use with SVCB records:

* SVCB, _coap, Section 7.1 of this document

The request for registration is deliberately not expressed at this point because it is yet to be revisited whether the creation of a "COAP" RR similar to the "HTTPS" RR would be preferable.

11. References

11.1. Normative References

[I-D.ietf-core-href]
Bormann, C. and H. Birkholz, "Constrained Resource Identifiers", Work in Progress, Internet-Draft, draft-ietf-core-href-23, 7 July 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-core-href-23>.

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

[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/rfc/rfc7252>.

[RFC8288] Nottingham, M., "Web Linking", RFC 8288, DOI 10.17487/RFC8288, October 2017, <https://www.rfc-editor.org/rfc/rfc8288>.

[RFC8742] Bormann, C., "Concise Binary Object Representation (CBOR) Sequences", RFC 8742, DOI 10.17487/RFC8742, February 2020, <https://www.rfc-editor.org/rfc/rfc8742>.

[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/rfc/rfc9052>.

[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/rfc/rfc9460>.

11.2. Informative References

[aliases] W3C, "Architecture of the World Wide Web, Section 2.3.1 URI aliases", n.d., <https://www.w3.org/TR/webarch/#uri-aliases>.

[cooluris] BL, T., "Cool URIs don’t change", n.d., <https://www.w3.org/Provider/Style/URI>.

[evossl] Baier, E., "The Evolution of SSL and TLS", 2 February 2015, <https://www.digicert.com/blog/evolution-of-ssl>.

[I-D.amsuess-core-coap-over-gatt] Amsüss, C., "CoAP over GATT (Bluetooth Low Energy Generic Attributes)", Work in Progress, Internet-Draft, draft-amsuess-core-coap-over-gatt-07, 25 September 2024, <https://datatracker.ietf.org/doc/html/draft-amsuess-core-coap-over-gatt-07>.

[I-D.amsuess-core-resource-directory-extensions] Amsüss, C., "CoRE Resource Directory Extensions", Work in Progress, Internet-Draft, draft-amsuess-core-resource-directory-extensions-11, 6 November 2024, <https://datatracker.ietf.org/doc/html/draft-amsuess-core-resource-directory-extensions-11>.

[I-D.amsuess-t2trg-onion-coap] Amsüss, C., Tiloca, M., and R. Hörglund, "Using onion routing with CoAP", Work in Progress, Internet-Draft, draft-amsuess-t2trg-onion-coap-03, 17 November 2024, <https://datatracker.ietf.org/doc/html/draft-amsuess-t2trg-onion-coap-03>.

[I-D.amsuess-t2trg-rdlink] Amsüss, C., "rdlink: Robust distributed links to constrained devices", Work in Progress, Internet-Draft, draft-amsuess-t2trg-rdlink-01, 23 September 2019, <https://datatracker.ietf.org/doc/html/draft-amsuess-t2trg-rdlink-01>.

[I-D.bormann-t2trg-slipmux] Bormann, C. and T. Kaupat, "Slipmux: Using an UART interface for diagnostics, configuration, and packet transfer", Work in Progress, Internet-Draft, draft-bormann-t2trg-slipmux-03, 4 November 2019, <https://datatracker.ietf.org/doc/html/draft-bormann-t2trg-slipmux-03>.

[I-D.ietf-cbor-edn-literals]
Bormann, C., "CBOR Extended Diagnostic Notation (EDN)", Work in Progress, Internet-Draft, draft-ietf-cbor-edn-literals-18, 7 July 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-cbor-edn-literals-18>.

[I-D.ietf-core-coap-dtls-alpn]
Lenders, M. S., Amsç«¯ss, C., Schmidt, T. C., and M. Wè¾°hlisch, "ALPN ID Specification for CoAP over DTLS", Work in Progress, Internet-Draft, draft-ietf-core-coap-dtls-alpn-04, 1 April 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-core-coap-dtls-alpn-04>.

[I-D.ietf-core-dns-over-coap]
Lenders, M. S., Amsç«¯ss, C., Gç«¯ndoãƒˆæ¬¡n, C., Schmidt, T. C., and M. Wãƒ†ã€hlisch, "DNS over CoAP (DoC)", Work in Progress, Internet-Draft, draft-ietf-core-dns-over-coap-16, 7 July 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-core-dns-over-coap-16>.

[I-D.ietf-lake-edhoc]
Selander, G., Mattsson, J. P., and F. Palombini, "Ephemeral Diffie-Hellman Over COSE (EDHOC)", Work in Progress, Internet-Draft, draft-ietf-lake-edhoc-23, 22 January 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-lake-edhoc-23>.

[I-D.ietf-lpwan-coap-static-context-hc]
Minaburo, A., Toutain, L., and R. Andreasen, "Static Context Header Compression (SCHC) for the Constrained Application Protocol (CoAP)", Work in Progress, Internet-Draft, draft-ietf-lpwan-coap-static-context-hc-19, 8 March 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-lpwan-coap-static-context-hc-19>.

[I-D.ietf-tls-esni]
Rescorla, E., Oku, K., Sullivan, N., and C. A. Wood, "TLS Encrypted Client Hello", Work in Progress, Internet-Draft, draft-ietf-tls-esni-25, 14 June 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-esni-25>.

[I-D.lenders-core-dnr]
Lenders, M. S., Amsãƒ†ã‚·ss, C., Schmidt, T. C., and M. Wãƒ†ã€hlisch, "Discovery of Network-designated OSCORE-based Resolvers: Problem Statement", Work in Progress, Internet-

Draft, draft-lenders-core-dnr-06, 7 July 2025, <https://datatracker.ietf.org/doc/html/draft-lenders-core-dnr-06>.

[I-D.silverajan-core-coap-protocol-negotiation] Silverajan, B. and M. Ocak, "CoAP Protocol Negotiation", Work in Progress, Internet-Draft, draft-silverajan-core-coap-protocol-negotiation-09, 2 July 2018, <https://datatracker.ietf.org/doc/html/draft-silverajan-core-coap-protocol-negotiation-09>.

[I-D.tiloca-core-oscore-capable-proxies] Tiloca, M. and R. Hテカglund, "OSCORE-capable Proxies", Work in Progress, Internet-Draft, draft-tiloca-core-oscore-capable-proxies-07, 10 July 2023, <https://datatracker.ietf.org/doc/html/draft-tiloca-core-oscore-capable-proxies-07>.

[I-D.tiloca-lake-app-profiles] Tiloca, M. and R. Hテカglund, "Coordinating the Use of Application Profiles for Ephemeral Diffie-Hellman Over COSE (EDHOC)", Work in Progress, Internet-Draft, draft-tiloca-lake-app-profiles-03, 21 October 2024, <https://datatracker.ietf.org/doc/html/draft-tiloca-lake-app-profiles-03>.

[lwm2m] OMA SpecWorks, "White Paper 寰Lightweight M2M 1.1", n.d., <https://omaspecworks.org/white-paper-lightweight-m2m-1-1/>.

[noproxy] Hu, S., "We need to talk: Can we standardize NO_PROXY?", 27 January 2021, <https://about.gitlab.com/blog/2021/01/27/we-need-to-talk-no-proxy/>.

[RFC1123] Braden, R., Ed., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, DOI 10.17487/RFC1123, October 1989, <https://www.rfc-editor.org/rfc/rfc1123>.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, DOI 10.17487/RFC2616, June 1999, <https://www.rfc-editor.org/rfc/rfc2616>.

[rfc3172] Huston, G., Ed., "Management Guidelines & Operational Requirements for the Address and Routing Parameter Area Domain ("arpa")", BCP 52, RFC 3172, DOI 10.17487/RFC3172, September 2001, <https://www.rfc-editor.org/rfc/rfc3172>.

[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/rfc/rfc3986>.

[RFC4648] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, DOI 10.17487/RFC4648, October 2006, <https://www.rfc-editor.org/rfc/rfc4648>.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <https://www.rfc-editor.org/rfc/rfc5952>.

[RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, DOI 10.17487/RFC6698, August 2012, <https://www.rfc-editor.org/rfc/rfc6698>.

[RFC7291] Boucadair, M., Penno, R., and D. Wing, "DHCP Options for the Port Control Protocol (PCP)", RFC 7291, DOI 10.17487/RFC7291, July 2014, <https://www.rfc-editor.org/rfc/rfc7291>.

[RFC7301] Friedl, S., Popov, A., Langley, A., and E. Stephan, "Transport Layer Security (TLS) Application-Layer Protocol Negotiation Extension", RFC 7301, DOI 10.17487/RFC7301, July 2014, <https://www.rfc-editor.org/rfc/rfc7301>.

[RFC7838] Nottingham, M., McManus, P., and J. Reschke, "HTTP Alternative Services", RFC 7838, DOI 10.17487/RFC7838, April 2016, <https://www.rfc-editor.org/rfc/rfc7838>.

[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/rfc/rfc8323>.

[RFC8552] Crocker, D., "Scoped Interpretation of DNS Resource Records through "Underscored" Naming of Attribute Leaves", BCP 222, RFC 8552, DOI 10.17487/RFC8552, March 2019, <https://www.rfc-editor.org/rfc/rfc8552>.

[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/rfc/rfc9176>.

[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/rfc/rfc9176>.

[RFC9200] Seitz, L., Selander, G., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "Authentication and Authorization for Constrained Environments Using the OAuth 2.0 Framework (ACE-OAuth)", RFC 9200, DOI 10.17487/RFC9200, August 2022, <https://www.rfc-editor.org/rfc/rfc9200>.

[RFC9461] Schwartz, B., "Service Binding Mapping for DNS Servers", RFC 9461, DOI 10.17487/RFC9461, November 2023, <https://www.rfc-editor.org/rfc/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/rfc/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/rfc/rfc9463>.

[rfc9476] Kumari, W. and P. Hoffman, "The .alt Special-Use Top-Level Domain", RFC 9476, DOI 10.17487/RFC9476, September 2023, <https://www.rfc-editor.org/rfc/rfc9476>.

[SUIB] "Router and IoT Vulnerabilities: Insecure by Design", 2021, <https://manysecured.net/wp-content/uploads/2022/09/ManySecured-SUIB-White-Paper.pdf>.

[w3address] BL, T., "W3 address syntax: BNF", 29 June 1992, <http://info.cern.ch/hypertext/WWW/Addressing/BNF.html#43>.

Appendix A. Change log

// This section is to be removed before publication.

Since draft-ietf-core-transport-indication-08:

* Explicitly become an SVCB mapping document for the CoAP schemes.

* Remove coaptransport parameter; instead, request ALPNs for remaining protocols.

* Point out issue about some metadata pertaining to transport choices and some metadata pertaining to the server itself.

* Acknowledge that there is structure (eg. in SVCB lookups) that is conceptually flattened.

* Sulkily accept DNS-over-CoAP reaching into target names as an exception in security requirements.

* Editorial updates.

Since draft-ietf-core-transport-indication-07:

* Update sections 1 and 2:

  - Work more explicitly with 7252’s resolution services.

  - List (coarsely) the information received from resolution services.

  - Homogenize treatment of proxy URIs and other resolution service outcomes.

  - Point out parallels to /etc/hosts.

* Phrase possible differences w/rt hop-to-hop vs. end-to-end encryption more carefully.

* SVCB: Rename edhoc-cred to cred.

* Rework literals following discussion around onion-coap.

* Point out the fates of appendices.

* Editorial fixes.

Since draft-ietf-core-transport-indication-06:

* Split introduction into terminology (with new definitions), goals and concepts.

* Add principle of operation into abstract, elevating SVCB and has-proxy to equally ranked sources of endpoint information.

* Restructure document to split overview and operations from the concrete methods of obtaining endpoints.

* Add is-unique-proxy SVCB parameter equivalent to has-unique-proxy relation.

* Remove _coaps service, describing _coap as applying to all CoAP transports.

* Add SVCB to abstract.

* Remove distracting text on URIs identifying transports/endpoints.

* Editorial changes.

* IANA considerations: Set change controller to IETF.

Since draft-ietf-core-transport-indication-05:

* Semantics for where a has-proxy applies were changed from "wherever there is a hosts relation" to "across the same origin".

  The hosts relation has received complaints about its complexity, and there were no strong voices in either direction during or after IETF119 when the question has been asked; going for the easier version.

* Use of SVCB is added as a section. Underscore prefixes are registered for CoAP, enabling the use of SVCB DNS records for applications that opt in to it (rather than processing it as an alternative history).

  While the alternative history section was appreciated during IETF119, the authors found it highly impractical to provide SVCB ground work without having the actual registrations (it would have worked only because DNS discovery acts on a separate _dns prefix anyway), and chose the consistent approach of allowing SVCB lookups.

* Material from the DNS and DNR for CoAP documents was moved in (and overhauled in the process):

  - Constructing CoAP requests from Service Parameters that did not result from a host name lookup is described.

- The coaptransport SVCB parameter is defined.

- SVCB hints for ACE/OAuth are defined.

* Section on how a host can tell that Uri-Host is optional was moved from Open Questions into a section.

  This had been around for ages, and gathering some more experience with the matter, looks like the obvious approach.

* Editorial:

  - Style for unallocated registrations changed from TBD to CPA

  - References updated

  - Tooling updates

  - Minor fixes

Since draft-ietf-core-transport-indication-04:

* Not just the scheme, but also the authority value influences the transport selection.

  - Add guidance section for new transports.

  - Point out that registerd names already can fan out to different addresses.

* Rephrase and simplify security considerations, especially by limiting unique proxying for TLS.

* Add recommendation to new scheme authors to use "coap"/"coaps" and let the resolution process guide the selection.

  - Remove proxy-schemes attribute from core.proxy because of its greatly reduced value.

* Update "Related work" appendix to cover SVCB instead of SRV records

* Rename to "Transport Indication", using "protocol" only for other protocols, in established phrases, or when referring to CoAP as a general protocol.

* Add note linking CoAP-over-WS's .well-known/coap to dohpath

* Remove OSCORE vs. unique-proxy open point

* EDHOC EAD: Describe response option content

* Editorial updates

Since draft-ietf-core-transport-indication-03:

* Added appendices on alternative history and Literals beyond IP addresses. The remaining document was not brought in sync with those new parts.

Since draft-ietf-core-transport-indication-02:

* Added EAD appendix, adjusted security considerations to match.

Since draft-ietf-core-transport-indication-01:

* Simplify same-host proxy behavior by referring to existing RFC7252 mandate.

* proxy-links= lookup: Refer to prior art.

Since draft-ietf-core-transport-indication-00:

* Add section on canonical URIs that are not necessarily reachable.

* Clarify that the the "hosts" relation is followed transitively.

* Cross reference with compatible multicast-notifications concept.

Since draft-amsuess-core-transport-indication-03:

* No changes (merely changing the name after WG adoption)

Since -02 (mainly processing reviews from Marco and Klaus):

* Acknowledge that ’coap://hostname/’ is not the proxy but a URI that (in a particular phrasing) is used to stand in for the proxy’s address (while it regularly identifies a resurce on the server)

* Security: Referencing traffic misdirection already in the first security block.

* Security: Add (incomplete) considerations for unique-proxy case.

* Narrow down "unique" proxy semantics to those properties used by the client, allowing unique proxies to be co-hosted with forward proxies.

* "Client picked proxies" clarified to merely illustrate how this is compatible with them.

* Use of "hosts" relation sharpened.

* Precision on how this does and does not consider changing transports.

* "Related work" section demoted to appendix.

* Add note on DTLS session resumption.

* Variable renaming.

* Various editorial fixes.

Since -01:

* Removed suggestion for generally trusted proxies; now stating that with (D)TLS, "a third party proxy can usually not satisfy [the security context propagation requirement]".

* State more clearly that valid cache entries for resources aliased through has-unique-proxy can be used.

* Added considerations for Multipath TCP.

* Added concrete suggestion and example for advertisement of general proxies.

* Added concrete suggestion for RD lookup extension that provides proxies.

* Minor editorial and example changes.

Since -00:

* Added introduction

* Added examples

* Added SCHC analogy

* Expanded security considerations

* Added guidance on choice of transport, and canonical addresses

* Added subsection on interaction with a Resource Directory

* Added comparisons with related work, including rdlink and DNS-SD sketches

* Added IANA considerations

* Added section on open questions

Appendix B. Related work and applicability to related fields

// This section is to be removed before submission to the IESG, with
// at least the MP-TCP section worked into corr-clar.

B.1. On HTTP

The mechanisms introduced here are similar to the Alt-Svc header of [RFC7838] in that they do not create different application-visible addresses, but provide dispatch through lower transport implementations.

In HTTP, different versions of the protocol (i.e., different transports) are distinguished using a protocol identifier equivalent to an ALPN. This works well because all relevant transports use transport layer security and thus can use ALPNs. In contrast, the currently specified CoAP transports predate ALPNs, and specified per-transport schemes, which enable the use of URIs that indicate transports.

To accommodate the message size constraints typical of CoRE environments, and accounting for the differences between HTTP headers and CoAP options, information is delivered once at discovery time.

Using the has-proxy and has-unique-proxy with HTTP URIs as the context is NOT RECOMMENDED; the HTTP provisions of the Alt-Svc header and ALPN are preferred.

B.2. Using DNS

DNS Service Binding resource records (SVCB RRs) described in [RFC9460] can carry many of the details otherwise negotiated using the proxy relations. In HTTP, they can be used in a way similar to Alt-Svc headers.

SVCB records were not specified when CoAP was specified for TCP.

If at any point SVCB records for CoAP are defined, name resolution produces a set of transport details that can be used immediately without the need for a has-proxy link. Explicit has-proxy links would still be relevant for third party advertised proxies.

B.3. Using names outside regular DNS

Names that are resolved through different mechanisms than DNS, or names which are defined within the scope of DNS but have no universally valid answers to A/AAAA requests, can be advertised using the relation types defined here and CoAP discovery.

In Figure 5, a server using a cryptographic name as described in [I-D.amsuess-t2trg-rdlink] is discovered and used.

```
Req: to [ff02::fd]:5683 on UDP
Code: GET
Uri-Path: /.well-known/core
Uri-Query: rel=has-proxy
Uri-Query: anchor=coap://nbswy3dpo5xxe3denbswy3dpo5xxe3de.ab.rdlink.arpa

Res: from [2001:db8::1]:5683
Content-Format: application/link-format
Payload:
<coap+tcp://[2001:db8::1]>;rel=has-unique-proxy;
  anchor="coap://nbswy3dpo5xxe3denbswy3dpo5xxe3de.ab.rdlink.arpa"


Req: to [2001:db8::1]:5683 on TCP
Code: GET
OSCORE: ...
Uri-Path: /sensors/temp
Observe: 0

Res: 2.05 Content
OSCORE: ...
Observe: 0
Payload:
39.1属C
```

Figure 5: Obtaining a sensor value from a local device with a global name

B.4. Multipath TCP

When CoAP-over-TCP is used over Multipath TCP and no Uri-Host option is sent, the implicit assumption is that there is aliasing between URIs containing any of the endpoints' addresses.

As these are negotiated within MPTCP, this works independently of this document's mechanisms. As long as all the server's addresses are equally reachable, there is no need to advertise multiple addresses that can later be discovered through MPTCP anyway. When advertisements are long-lived and there is no single more stable address, several available addresses can be advertised (independently of whether MPTCP is involved or not). If a client uses an address that is merely a proxy address (and not a unique proxy address), but during MPTCP finds out that the network location being accessed is actually an MPTCP alternative address of the used one, the client MAY forego sending of the Proxy-Scheme and Uri-Path option.

[This follows from multiple addresses being valid for that TCP connection; at some point we may want to say something about what that means for the default value of the Uri-Host option -- maybe something like "has the default value of any of the associated addresses, but the server may only enable MPTCP if there is implicit aliasing between all of them" (similar to OSCORE's statement)?]

[TBD: Do we need a section analog to this that deals with (D)TLS session resumption in absence of SNI?]

Appendix C. Open Questions / further ideas

// This section is to be either converted into concrete guidance, or
// removed before submission to the IESG.

* Advertising under a stable name:

  If a host wants to advertise its host name rather than its IP address during multicast, how does it best do that?

  Options, when answering from 2001:db8::1 to a request to ff02::fd are:

<coap://myhostname/foo>,...,
<coap://[2001:db8::1]>;rel=has-unique-proxy;anchor="coap://myhostname"

  which is verbose but formally clear, and

</foo>,...,
<coap://[2001:db8::1]>;rel=has-unique-proxy;anchor="coap://myhostname"

  which is compatible with unaware clients, but its correctness with respect to canonical URIs needs to be argued by the client, in this sequence

- understanding the has-unique-proxy line,

- understanding that the request that went to 2001:db8::1 was really a Proxy-Scheme/Uri-Host-elided version of a request to coap://myhostname, and then

- processing any relative reference with this new base in mind.

(Not that it'd need to happen in software in that sequence, but that's the sequence needed to understand how the /foo here is really coap://myhostname/foo).

If CoRAL is used during discovery, a base directive or reverse relation to has-unique-proxy would make this easier.

Appendix D. EDHOC EAD for verifying legitimate proxies

// This section is to be moved into another document, possibly
// groupcomm-proxy.

This document sketches an extension to [I-D.ietf-lake-edhoc] that informs the server of the public address the client is using, allowing it to detect undesired reverse proxies.

[This section is immature, and written up as a discussion starting point. Further research into prior art is still necessary.]

The External Authorization Data (EAD) item with name "Proxy CRI", label 24-CPA, is defined for use with messages 1, 2 and 3.

A client can set this label in uncritical form, followed by a CRI ([I-D.ietf-core-href]) that is CBOR-encoded in a byte string as a CBOR sequence. The transport indicated by the URI is the proxy the client chose from information advertised about the server.

If a server can not determine its set of legitimate proxies, it ignores the option (as does any EDHOC implementation that is unaware of it).

If it recognizes the CRI as belonging to a legitimate proxy, it places the empty label in its non-critical form in the next message to confirm the proxy choice. Otherwise, it places the label in its critical form, either empty or containing a recommended CRI. The client may then decide to discontinue using the proxy, or to use more extensive padding options to sidestep the attack. Both the client and the server may alert their administrators of a possible traffic misdirection.

[While using an EDHOC EAD is suitable for connection setup, such a mechanism may also be useful at a later time, e.g. to re-check a server's address after a name change; establishing an equivalent CoAP option is being considered, also oin light of the discussion around https://github.com/core-wg/corrclar/pull/40 and https://github.com/core-wg/groupcomm-proxy/issues/3.]

Appendix E. Literals beyond IP addresses

[This section is placed here preliminarily: After initial review in CoRE, this may be better moved into a separate document aiming for a wider IETF audience.]

E.1. Motivation for new literal-ish names

IP literals were part of URIs from the start [w3address]. Initially, they were equivalent to host names in their expressiveness, save for their inherent difference that the former can be used without a shared resolver, and the latter can be switched to a different network address.

This equivalence got lost gradually: Certificates for TLS (its precursor SSL has been available since 1995 [evossl]) have only practically been available to host names. The Host header introduced in HTTP 1.1 Section 14.23 of [RFC2616] introduced name based virtual hosting in 1999. DANE [RFC6698], which provides TLS public keys augmenting the or outside of the public key infrastructure, is only available for host names resolved through DNSSEC. SVCB records [RFC9460] introduced in 2023 allow starting newer HTTP transports without going through HTTP/1.1 first, enables load balancing, fail-over, and enable Encrypted Client Hello -- again, only for host names resolved through DNS.

This document proposes an expression for the host component of a URI that fills that gap. Note that no attempt is yet made to register service.arpa in the .ARPA Zone Management; that name is used only for the purpose of discussion.

```
// The structure and even more the syntax used here is highly
// preliminary. They serves to illustrate that the desired
// properties can be obtained, and do not claim to be optimal. As
// one of many aspects, they are missing considerations for
// normalization and for internationalization.
```

E.2. Structure of service.arpa

Names under service.arpa are structured into an ordered list of key-value component pairs, and the common suffix service.arpa.

These pairs represent the very items also produced by Section 2.1. In the current version, they can _not_ express multiple entries with different structures. They can express different entries, but any repeated items (e.g. different ALPNs and IP addresses) only produce their product (i.e., no "TCP on this or that address, TLS on another").

Keeping with the style of DNS (least significant component first), pairs are expressed with their data a first label, followed by the key in a separate label. Data items ending with a "-" character are concatenated with their previous label to avoid overflowing the 63-octet limit of DNS (even though those do not wind up in DNS serialization).

For example, "world.hello-.label.8080.port" contains, in that order, information about port 8080 and the label "helloworld".

Initial component types are:

* "6": The value encodes an IPv6 address in [RFC5952] format, with the colon character (":") replaced with a dash ("-").

  It indicates an address of a host providing the service.

  If any address information is present, a client SHOULD use that information to access the service.

* "4": The value encodes an IPv4 address in dotted decimal format [RFC1123], with the dot character (".") replaced with a dash ("-").

  It indicates an address of a host providing the service.

* "tlsa": The data of a DNS TLSA record [RFC6698] encoded in base32 [RFC4648].

  Depending on the usage, this describes TLS credentials through which the service can be authenticated.

  If present, a client MUST establish a secure connection, and MUST fail the connection if the TLSA record’s requirements are not met.

* "cred", "edhoc-info", "oauth-info": SvcbParams in base32 encoding of their wire format.

* "alpn": The ALPN(s) in hexadecimal encoding, separated by dashes.

Due to [RFC3986]’s rules, all components are case insensitive and canonically lowercase.

Note that while using the IPvFuture mechanism of [RFC3986] would avoid compatibility issues around the 63 character limit and some of the character restrictions, it would not resolve the larger limitation of case insensitivity.

E.3. Processing service.arpa

A client accessing a resource under a service.arpa name does not consult DNS, but obtains information equivalent to the results of a DNS query from parsing the name.

DNS resolvers should refuse to resolve service.arpa names. (They would have all the information needed to produce sensible results, but operational aspects would need a lot of consideration; future versions of this document may revise this).

E.4. Examples

TBD: For SvcParams, the examples are inconsistent with the base32 recommendation; they serve to explore the possible alternatives.

* http://683263.alpn.2001-db8--1.6.service.arpa/ -- The server is reachable on 2001:db8::1 using HTTP/2 (h2c is 683263 in hex)

* https://amaqckrkfivcukrkfivcukrkfivcukrkfivcukrkfivcukrkfivcukrk.tlsa.service.arpa/ -- No address information is provided, the client needs to resort to other discovery mechanisms. Any server is eligible to serve the resource if it can present a (possibly self-signed) certificate whose public key SHA256 matches the encoded value.

* coap://434f4150.alpn.2001-db8--1.6.rai3ouj4a.ueekcandaeasabbblaqlxq2jmbjg5jgtf2kazljkenaurxocc6i2ckx3zowjgy-.cred.service.arpa/ -- The server is reachable using CoAP over TCP with any security mechanism at 2001:db8::1, and the service is identifiable by the use of a KCCS credential describing an X25519 public key (which is currently only usable with EDHOC).

* coap://rai3ouj4a.ueekcandaeasabbblaqlxq2jmbjg5jgtf2kazljkenaurxocc6i2ckx3zowjgyr-.cred.service.arpa/ -- The same server without any discoverability hints; it is up to the client to discover a (possibly short-lived) connection opportunities to the server identified by that key.

Appendix F. Acknowledgements

This document heavily builds on concepts explored by Bill Silverajan and Mert Ocak in [I-D.silverajan-core-coap-protocol-negotiation], and together with Ines Robles and Klaus Hartke inside T2TRG.

[TBD: reviewers Marco Klaus]

Authors' Addresses

Christian Amsüss
Austria
Email: christian@amsuess.com

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