CoRE Working Group
Internet-Draft
Intended status: Informational
Expires: January 9, 2017

B. Silverajan
TUT
M. Ocak
Ericsson

July 8, 2016

CoAP Protocol Negotiation
draft-silverajan-core-coap-protocol-negotiation-03

Abstract

CoAP has been standardised as an application-level REST-based protocol. When multiple transport protocols exist for exchanging CoAP resource representations, this document introduces a way forward for CoAP endpoints as well as intermediaries to agree upon alternate transport and protocol configurations as well as URIs for CoAP messaging.

Table of Contents

1. Introduction

The Constrained Application Protocol (CoAP) [RFC7252] allows clients, origin servers and proxies, to exchange and manipulate resource representations using REST-based methods using UDP or DTLS. CoAP messaging is however being extended to use other alternative underlying transports. These include reliable transports such as TCP, WebSockets and TLS. In addition, the use of SMS as a CoAP transport remains a possibility for simple communication in cellular networks.

When CoAP-based endpoints and proxies possess the ability to perform CoAP messaging over multiple transports, significant benefits can be obtained if communicating client endpoints can discover that multiple transport bindings may exist on an origin server over which CoAP resources can be retrieved. This allows a client to understand and possibly subsitute a different transport protocol configuration for the same CoAP resources on the origin server, based on the preferences of the communicating peers. Inevitably, if two CoAP endpoints reside in distinctly separate networks with orthogonal transports, a CoAP proxy node is needed between the two networks so that CoAP Requests and Responses can be exchanged properly.

A URI in CoAP, however, serves two purposes simultaneously. It firstly functions as a locator, by specifying the network location of the endpoint hosting the resource, and the underlying transport used by CoAP for accessing the resource representation. It secondly identifies the name of the specific resource found at that endpoint together with its namespace, or resource path. A single CoAP URI cannot be used to express the identity of the resource independently of alternate underlying transports or protocol configuration. Multiple URIs can result for a single CoAP resource representations if:

- the authority components of the URI differ, owing to the same physical host exposing several network endpoints. For example, "coap://example.org/sensors/temperature" and "coap://example.net/sensors/temperature"

- the scheme components of the URI differ, owing to the origin server exposing several underlying transport alternatives. For example, "coap://example.org/sensors/temperature" and "coap+tcp://example.org/sensors/temperature"

- the path components of the URI differ, should an origin server also allow alternative transport endpoint such as the WebSocket protocol, to be expressed using the path. For example, "coap://example.org/sensors/temperature" and "coap+ws://example.org/ws-endpoint/sensors/temperature"

Without a priori knowledge, clients would be unable to ascertain if two or more URIs provided by an origin server are associated to the same representation or not. Consequently, a communication mechanism needs to be conceived to allow an origin server to properly capture the relationship between these alternate representations or locations and then subsequently supply this information to clients. This also goes some way in limiting URI aliasing [WWWArchv1].

In order to support CoAP clients, proxies and servers wishing to use CoAP over multiple transports, this draft proposes the following:

- A means for CoAP clients to interact with an origin server's CoRE resource directory interface to discover alternative transports and links describing alternate locations of CoAP resources.

- An ability for servers to convey the names of supported CoAP transports to requesting clients, in order of preference, as well as any optional lifetime values associated with them.

- o A new link format attribute as well as a new link relation type that together enable an origin server to serve a resource from other protocol configurations or endpoints.

2. Aim

The following simple scenarios aim to better portray how CoAP protocol negotiation benefits communicating nodes

2.1. Overcoming Middlebox Issues

Discovering which transports are available is important for a client to determine the optimal alternative to perform CoAP messaging according to its needs, particularly when separated from a CoAP server via a NAT. It is well-known that some firewalls as well as many NATs, particularly home gateways, hinder the proper operation of UDP traffic. NAT bindings for UDP-based traffic do not have as long timeouts as TCP-based traffic.

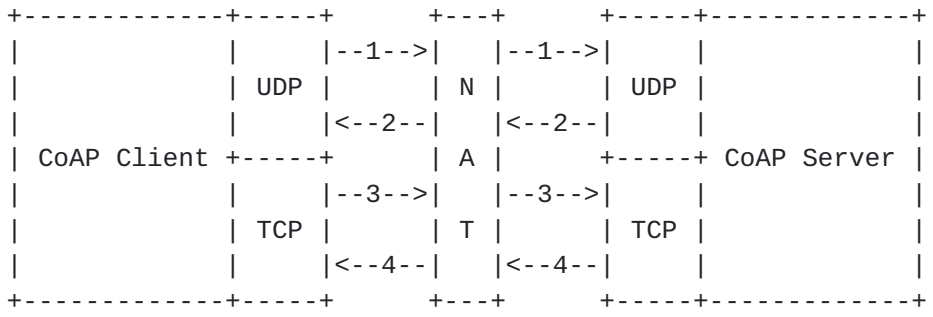

Figure 1: CoAP Client initially accesses CoAP Server over UDP and then switching to TCP

Figure 1 depicts such a scenario, where a CoAP client uses UDP initially for accessing a CoAP Server, and engages in discovering alternative transports offered by the server. The client subsequently decides to use TCP for CoAP messaging instead of UDP to set up an Observe relationship for a resource at the CoAP Server, in order to avoid incoming packets containing resource updates being discarded by the NAT.

2.2. Better resource caching and serving in proxies

Figure 2 outlines a more complex example of intermediate nodes such as CoAP-based proxies to intelligently cache and respond to CoAP or HTTP clients with the same resource representation requested over alternative transports or server endpoints.

In this example, a CoAP over WebSockets client successfully obtains a response from a CoAP forward proxy to retrieve a resource representation from an origin server using UDP, by supplying the CoAP server's endpoint address and resource in a Proxy-URI option. Arrow 1 represents a GET request to "coap+ws://proxy.example.com" which subsequently retrieves the resource from the CoAP server using the URI "coap://example.org/sensors/temperature", shown as arrow 2.

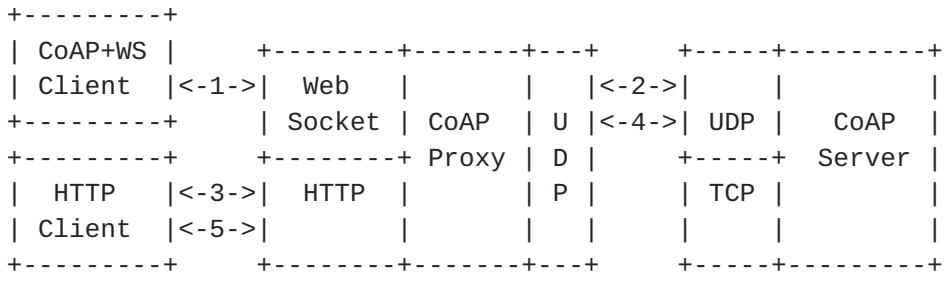

Figure 2: Proxying and returning a resource's alternate cached representations to multiple clients

Subsequently, assume an HTTP client requests the same resource, but instead specifies a CoAP over TCP alternative URI instead. Arrow 3 represents this event, where the HTTP client performs a GET request to "http://proxy.example.com/coap+tcp://example.org/sensors/temperature". When the proxy receives the request, instead of immediately retrieving the temperature resource again over TCP, it first verifies from the CoAP server whether the cached resource retrieved over UDP is a valid equivalent representation of the resource requested by the HTTP client over TCP (arrow 4). Upon confirmation, the proxy is able to supply the same cached representation to the HTTP client as well (arrow 5).

2.3. Interaction with Energy-constrained Servers

Figure 3 illustrates discovery and communication between a CoAP client and an energy-constrained CoAP Server. Such a server aims at conserving its energy unless a need arises otherwise. The figure

depicts the server maintaining its communication in a low-power state by listening only for incoming SMS messages while disabling communication on radio interfaces requiring greater energy. This is depicted as the server's initial state in the figure, showing an active SMS endpoint and a disabled or dormant UDP interface.

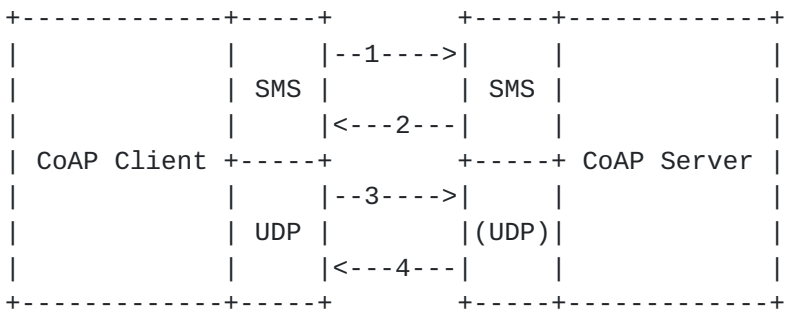

Figure 3: CoAP client interacting over SMS to discover a server's IP-based endpoint

A CoAP client wishing to perform CoAP operations can query a CoAP server for available transports via SMS, as shown in arrow 1. Upon reception of the message, should the server have its radio and IP interface up, it can send an SMS response containing the location of the CoAP IP endpoint and supported transports. Alternatively, the incoming SMS can be also used by the server as a triggering event to temporarily power up its radio interface so that UDP or other transport-based CoAP communication can instead be employed, and likewise provide this information in its response. This is depicted as arrow 2. Subsequently, low latency IP-based CoAP communication can occur between the endpoints as shown by arrows 3 and 4.

3. Node Types based on Transport Availability

In [RFC7228], Tables 1, 3 and 4 introduced classification schemes for devices, in terms of their resource constraints, energy limitations and communication power. For this document, in addition to these capabilities, it seems useful to additionally identify devices based on their transport capabilities.

Name	Transport Availability
T0	Single transport
T1	Multiple transports, with one or more active at any point in time
T2	Multiple active and persistent transports at all times

Table 1: Classes of Available Transports

Type T0 nodes possess the capability of exactly 1 type of transport channel for CoAP, at all times. These include both active and sleepy nodes, which may choose to perform duty cycling for power saving.

Type T1 nodes possess multiple different transports, and can retrieve or expose CoAP resources over any or all of these transports. However, not all transports are constantly active and certain transport channels and interfaces could be kept in a mostly-off state for energy-efficiency, such as when using CoAP over SMS (refer to section 2.1)

Type T2 nodes possess more than 1 transport, and multiple transports are simultaneously active at all times in a persistent manner. CoAP proxy nodes which allow CoAP endpoints from disparate transports to communicate with each other, are a good example of this.

4. New Link Attribute and Relation types

A CoAP server wishing to allow interactions with resources from multiple locations or transports can do so by specifying the Transport Type "tt" link attribute, which is an opaque string. Multiple transport types can be included in the value of this parameter, each separated by a space. In such cases, transport types appear in a prioritised list, with the most preferred transport type by the CoAP server specified first and the lowest priority transport type last.

At the same time, each transport type supported by the server is also described with an "altloc" link relation type. The "altloc" relation type specifices a URI (containing the URI scheme, authority and

optionally path) providing an alternate endpoint location up to but not including the resource path of a representation.

Each URI specified by "altloc" link relation type can also have an active lifetime value described with "al" link extension, which is an integer showing the active lifetime in seconds. The "al" link extension specifies how long the CoAP server will respond to the specified URI in "altloc" relation type.

Both "tt" and "altloc" are optional CoAP features. If supported, they occur at the granularity level of an origin server, ie. they cannot be applied selectively on some resources only. Therefore "altloc" is always anchored at the root resource ("/"). The "al" link attribute, while also being optional, exists at the granularity of each transport protocol used. When it is absent, it is assumed that the transport protocol is persistent.

Additionally, the "tt" and "al" link attributes as well as the "altloc" relation type can be ignored by unsupported CoAP clients.

5. Observing Transport Types and Resource Representations

A CoAP client interested in being notified of changes in an origin server's transport protocols for CoAP, can choose to do so with an Observe relationship [RFC7641]. The client registers its interest on the available active transports by setting the Observe option with a GET to ".well-known/core" on a CoAP server, with a client-specified parameter value for "tt" as depicted in Figure 4. Updates on the active transports will be sent to the CoAP client as CoAP notifications.

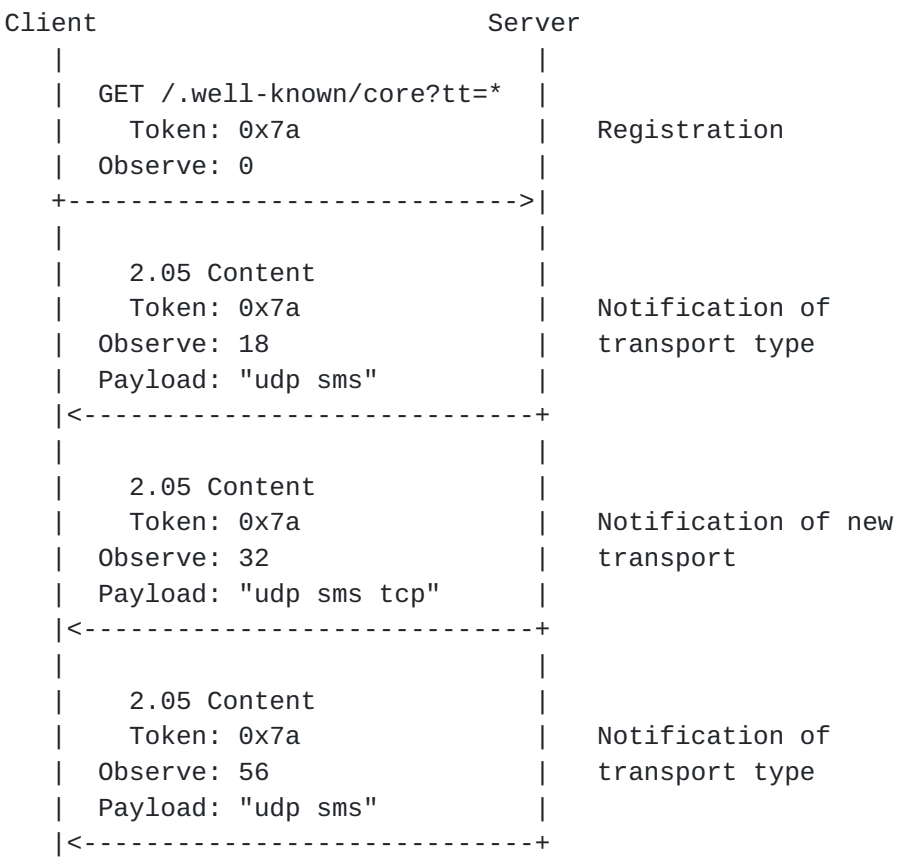

Figure 4: CoAP client observing .well-known/core for all transport types

Observe relationships between a CoAP client and a CoAP server must conform to established norms specified in [RFC7641]. Subsequent notifications are considered to simply be additional responses to the original client GET request. Therefore, should a client subsequently switch to a different transport protocol (such as from UDP to TCP), it is the responsibility of the client to deregister its interest beforehand or cancel its interest as specified in section 3.6 of [RFC7641]. No assumptions of session continuation should be made and the client should instead re-register its interest using the new transport, either actively, or upon the Max-Age of a stored representation being exceeded at the client.

A server can also prevent notifications to be perpetually sent to a client on a previous transport, by using confirmable CoAP messages for responses. This allows the server to remove an unresponsive client from its list of interested observers.

6. Examples

Figure 5 shows a CoAP server returning all transport types and the alternate resource locations to a CoAP client performing a CoAP Request to ./well-known/core

In this case, the server supplies two different locations to interact with resources using CoAP over TCP i.e. the resources given in the CoAP response are available from multiple hosts with different entry points and transport layer security.

At the same time, the path to the WebSocket endpoint is provided in addition to the FQDN of the server, for using CoAP over WebSockets, exemplifying the ability to separate a CoAP resource path from a web-based CoAP endpoint path in a URI.

```
REQ: GET /.well-known/core

RES: 2.05 Content
</sensors>;ct=40;title="Sensor Index", tt="tcp ws sms",
</sensors/temp>;rt="temperature-c";if="sensor",
</sensors/light>;rt="light-lux";if="sensor",
<coap+tcp://server.example.com/>;rel="altloc",
<coaps+tcp://server.example.net/>;rel="altloc",
<coap+ws://server.example.com/ws-endpoint/>;rel="altloc",
<coap+sms://001234567>;rel="altloc"
```

Figure 5: Example of Server response

Figure 6 shows a CoAP client actively soliciting a CoAP server for all supported transport types and protocol configurations.

```
REQ: GET /.well-known/core?tt=*

RES: 2.05 Content
</sensors>;tt="tcp sms ws"
<coap+tcp://server.example.com/>;rel="altloc",
<coap+tcp://server.example.net/>;rel="altloc",
<coap+ws://server.example.com/ws-endpoint/>;rel="altloc",
<coap+sms://001234567/>;rel="altloc"
```

Figure 6: CoAP client discovering transports supported by a CoAP server.

Figure 7 shows a CoAP client explicitly soliciting support for a specific transport type using a query filter parameter.

```
REQ: GET /.well-known/core?tt=sms

RES: 2.05 Content
</sensors>;tt="tcp sms ws"
<coap+sms://001234567/>;rel="altloc"
```

Figure 7: CoAP client looking for a specific transport to use with a CoAP server.

Figure 8 shows a CoAP client making a CoAP over SMS request to an energy-constrained CoAP server, explicitly soliciting support for UDP-based communication by using a query filter parameter. The server temporarily activates its UDP interface, responds with the location of the UDP endpoint and also provides the expected lifetime of the transport, which in this case is 120 seconds.

```
REQ: GET /.well-known/core?tt=udp

RES: 2.05 Content
</sensors>;tt="udp sms"
<coap://server.example.com/>;rel="altloc";al=120
```

Figure 8: CoAP client using CoAP over SMS to discover UDP-based address and transport lifetime.

7. IANA Considerations

This document requests the registration of a new link relation type "altloc".

Relation name:
altloc

Description:
Identifies an alternate CoAP endpoint location for a resource.

Reference:
This document.

8. Security Considerations

When multiple transports, locations and representations are used, some obvious risks are present both at the origin server as well as by requesting clients.

An energy-constrained node exposing its resource representations using CoAP over SMS, but subsequently enabling its IP interface on-demand, can be subjected to denial-of-sleep as well as battery draining attacks by attackers. The risk can be somewhat mitigated at the server by strict requirements on the active lifetime of IP-based communication as well as restricting which clients are allowed to request for IP-based communication and referring other incoming requests to a caching proxy instead.

When a client is presented with alternate URIs for retrieving resources, it presents an opportunity for attackers to mount a series of attacks, either by hijacking communication and masquerading as an alternate location or by using a man-in-the-middle attack on TLS-based communication to a server and redirecting traffic to an alternate location. A malicious or compromised server could also be used for reflective denial-of-service attacks on innocent third parties. Moreover, clients may obtain web links to alternate URIs containing weaker security properties than the existing session.

9. Acknowledgements

Thanks to Klaus Hartke for comments and reviewing this draft, and Teemu Savolainen for initial discussions about protocol negotations and lifetime values.

10. References

10.1. Normative References

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

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <http://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, <http://www.rfc-editor.org/info/rfc7641>.

10.2. Informative References

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

[WWWArchv1] http://www.w3.org/TR/webarch/#uri-aliases, "Architecture of the World Wide Web, Volume One", December 2004.

Appendix A. Change Log

A.1. From -02 to -03

Added new author

Rewrite of "Introduction" section

Added new Aims Section

Added new Section on Node Types

Introduced "al" Active Lifetime link attribute

Added new Section on Observing transports and resources

Security and IANA considerations sections populated

A.2. From -01 to -02

Freshness update.

A.3. From -00 to -01

Reworked "Introduction" section, added "Rationale", and "Goals" sections.

Authors' Addresses

Bilhanan Silverajan
Tampere University of Technology
Korkeakoulunkatu 10
FI-33720 Tampere
Finland

Email: bilhanan.silverajan@tut.fi

Mert Ocak
Ericsson
Hirsilantie 11
02420 Jorvase
Finland

Email: mert.ocak@ericsson.com