Publish Option for CoAP
draft-fossati-core-publish-option-02

Abstract

This memo defines the Publish Option for the Constrained Application Protocol (CoAP). This Option is used by a CoAP Endpoint to control the authority delegation of one of its resources to another Endpoint. All the phases of the authority delegation process (setup, renewal, cancellation) are controlled by a simple RESTful protocol.

This memo also introduces the ‘proxies’ Web Linking relation type, to be used by a CoAP Proxy to explicitly advertise the resources that it can serve from its cache, or by forwarding the Client request upstream (either to the origin, or to another Proxy).

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 http://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 April 23, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

This memo defines the Publish Option for the Constrained Application Protocol [I-D.ietf-core-coap].

The Publish Option is used by a sleepy Endpoint (SEP) to temporarily delegate the authority of one of its resources to another, always on, Endpoint. The delegated Endpoint is typically a Proxy, though it could be an Endpoint with no other special network role.

The SEP is given a simple RESTful messaging protocol that enables the setup, renewal and cancellation of the authority transfer. The whole process is driven by the SEP, which may actually never need to listen or to keep any state.
This memo also introduces the ‘proxies’ Web Linking [RFC5988] relation type.

This new relation, which complements the default ‘hosts’ relation defined in [RFC6690], can be used by a CoAP Proxy to explicitly advertise the resources that it can serve either from cache or by forwarding the Client request upstream.

The ‘proxies’ relation works in concert with the Publish Option to enable SEP discovery even while SEP is off-line.

1.1. Requirements Language and Motivation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

The terms Client, Proxy, Server, and Endpoint are to be interpreted as described in [I-D.ietf-core-coap].

This memo reuses the terminology introduced in [I-D.rahman-core-sleepy-problem-statement], and aims at meeting the objectives stated in its Section 4 via an entirely in-protocol solution.

2. Publish Option

The Publish Option enables a SEP to temporarily (i.e. for a specified "lease" time) delegate the authority of one of its hosted resources to another Endpoint.

<table>
<thead>
<tr>
<th>No.</th>
<th>C</th>
<th>U</th>
<th>N</th>
<th>R</th>
<th>Name</th>
<th>Format</th>
<th>Length</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>2n+1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>-</td>
<td>Publish</td>
<td>uint</td>
<td>1</td>
<td>(none)</td>
</tr>
</tbody>
</table>

2.1. Publishing a Resource

The SEP publishes one of its hosted resources, specified by the enclosed Proxy-URI, by making a PUT to the Proxy with a Publish Option attached. The Publish Option value specifies the CoAP methods that Clients are allowed to use on the resource (see Section 2.4).

The example in Figure 1 shows a delegation where the GET and PUT methods are allowed, while POST and DELETE are explicitly prohibited, meaning that the resource can only be read and updated by Clients.
The Proxy, which is voluntarily entrusted by the resource owner to act as the delegated origin for the "lease" time specified by Max-Age, replies with a 2.01 if the authority transfer succeeds. An exact duplicate of the submitted representation is created, and from now on it can be accessed using the original URI provided that Clients request it through the delegated Proxy. If the Publish operation isn’t successful (e.g. because the Proxy does not support Publish), then the origin transfer fails, and an appropriate response code is returned (e.g. 4.02 Bad Option).

An ETag MAY be supplied as a further metadatum to be included in responses involving the published representation. If no Max-Age is given, a default of 3600 seconds MUST be assumed. The Max-Age value, either implicit or explicit, determines the lifetime of the origin delegation. When Max-Age is elapsed, the Proxy MUST delete the published resource value and fall back to its usual proxying function.

The Publish Option is critical, and MUST NOT be present in a response. If the Proxy does not recognize it, a 4.02 (Bad Option) MUST be returned to the Client. If the Option value is not correctly formatted (see Section 2.4), a 4.00 (Bad Request) MUST be returned to the Client. The Publish Option is not Safe-to-Forward, and neither is a Cache-Key.

Since the 2.01 is emitted, and for the duration of the delegation, any Client wishing to access the resource can do so by making a Proxy-URI request to the Proxy, which shall then serve the resource off from its own storage.

An interesting outcome of this communication strategy is that the SEP may really never need to listen on its radio interface. However, ignoring the response status code from Proxy is not a safe practice and is not encouraged.
Upon publishing, the Proxy MUST save the identity of the publishing SEP, and MUST use it to correctly authorise "maintenance" operations such as renewal or cancellation of the published resource. The SEP identity MUST be kept for the whole duration of the delegation (including any associated renewal) and can be forgotten as soon as the delegation vanishes either implicitly or explicitly.

2.2. Updating a Resource

In order to update the delegated resource state or to just extend the lease period, the SEP shall send basically the same request (except possibly for the representation value and the associated ETag) to the Proxy, which in turn replies with a 2.04 Changed status code in case the update operation succeeds. If the operation fails, e.g. because the request comes from an Endpoint different from the publishing SEP, a suitable status code is returned (e.g. 4.01 Unauthorized).

```
P         S
|   PUT   | Proxy-URI: coap://sleepy.example.org/res
|<--------+ Publish: 0x60
|r        | Content-Format: text/plain
|         | ETag: 0xdcba
|         | Max-Age: 1200
| 2.04    |
|--------->|
```

Figure 2

2.3. Unpublishing a Resource

The delegation of a given resource can be explicitly revoked by the SEP at any time before the lease time expires, by issuing a DELETE request to the Proxy hosting the resource duplicate with a Publish Option with value 0x00.

```
P         S
|  DELETE  | Proxy-URI: coap://sleepy.example.org/res
|<---------+ Publish: 0x00
|         |
| 2.02    |
|--------->|
```

On successful deletion of the delegation, a 2.02 Deleted response code is returned by the Proxy. On error a suitable status code is returned.
2.4. Value Format

The Publish Option consists of a single byte having the following layout:

```
0 1 2 3 4 5 6 7
+-------+-------+
|   CRUD |   0000 |
+-------+-------+
```

Each of the higher 4 bits is a flag field indicating whether the associated CoAP method (respectively: POST, GET, PUT and DELETE) is allowed on the published resource. The lower 4 bits are reserved and MUST be set to 0.

The 0x00 value is used to explicitly revoke the delegation (see Section 2.3.) and MUST NOT be used for any other purpose of the Option.

If the delegated Proxy receives a request for the published resource with a method that is not compatible with the mask supplied by the SEP, it MUST respond with a 4.05 (Method Not Allowed) response code.

3. Discovery

3.1. The ‘proxies’ Relation Type

The new ‘proxies’ Web Linking [RFC5988] relation type is meant to signify that the target resource carried by the link, which MUST be identified by an absolute URI, is reachable through a Proxy-URI request made to the anchored Origin (i.e. the Proxy).

(Note that we need to specify the Proxy through an explicit anchor, thus increasing the verbosity of the link value, because of the way the context URI override rules are defined in Section 2.1 of [RFC6690]. In fact, absent an explicit anchor, rule (b) would set the context to the SEP origin, which is definitely not what we want.)

3.1.1. Examples

```
 o C discovers resources that P "proxies":

 P         C
 | GET      | Uri-Path: .well-known
 |----------| Uri-Path: core
 |          | Uri-Query: rel="proxies"
 | 2.05     |
```
C GET’s a "proxied" resource from P:

```
+--------> | <coap://sleepy.example.org/res>;  
|         |     anchor="coap://proxy.example.org/";  
|         |     rel="proxies",  
|         | <...  
```

The ‘proxies’ relation is orthogonal to the Publish Option, so it’s up to P to decide whether to serve coap://sleepy.example.org/res from its store/cache, or to forward the request to the origin at coap://sleepy.example.org.

### 3.2. Adjusting Link-Format Attributes

#### 3.2.1. Implicitly

The resource metadata are implicitly extracted from the published representation. Basically, the Proxy works out the ct and sz attributes by inspecting Content-Format and the request payload size.

#### 3.2.2. Explicitly

The resource metadata are explicitly published to the same Proxy-URI used for the sibling resource, either in a separate request/response cycle (see Figure 3),

```
P | PUT | Proxy-URI: coap://sleepy.example.org/res  
| <--------+ Publish: 0x60  
| <meta> | Content-Format: application/link-format  
| 2.01  
| +-------->
```

Figure 3
or atomically, within the same Publish operation, e.g. by using the Multipart Content-Format to aggregate one (or even more than one) representation(s) together with the application/link-format entry (see Figure 4).

```
P | PUT  | Proxy-URI: coap://sleepy.example.org/res
|<--------+ Publish: 0x60
| [mp]    | Content-Format: application/multipart+publish
|         | ETag: 0xabcd
|         | Max-Age: 1200
| 2.01    | +-------->
```

Figure 4

Note that the former (Figure 3) is non-atomic, and limited to only one representation of the resource; the latter (Figure 4) is atomic and supports multiple Content-Format’s for the published resource.

4. Acknowledgements

Thanks to Bruce Nordman and Matthieu Vial.

5. IANA Considerations

The following entries are added to the CoAP Option Numbers registry:

```
<table>
<thead>
<tr>
<th>Number</th>
<th>Name</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>2n+1</td>
<td>Publish</td>
<td>RFC XXXX</td>
</tr>
</tbody>
</table>
```

This memo registers the new "proxies" Web Linking relation type as per [RFC5988].

Relation Name: proxies

Description: the target is the absolute URI of a resource proxied by the Origin stated in the anchor.

Reference: this memo

Notes: This relation is used in CoRE where links are retrieved as a "/.well-known/core" resource representation.
6. Security Considerations

This section identifies Threats (T) and related countermeasures (C).

- **T:** cache poisoning.  
  **C:** use strong auth to identify SEP.

- **T:** unauthorized update or de-registration  
  **C:** strong auth to identify SEP.

- **T:** Proxy resources’ exhaustion.  
  **C:** use strong auth to identify SEP + quota limit.

- **T:** local state loss.  
  **C:** cache redundancy.

- **T:** Inject fake copies of the resource by a 3rd party.  
  **C:** use delegation scheme that bundles the identities of the SEP and the Proxy, together with the resource being delegated. A third party must be able to verify SEP and Proxy identities, maybe offline, and check the resource fingerprint.

6.1. Securing the Delegation

(Sketch) SEP signs the identity of the delegated Proxy and a fingerprint of the resource (both data and meta), and bundles it up with the resource itself, maybe in a MultiPart envelope (TBD define signed Content-Format). Client verifies the resource is indeed from the SEP by checking the signature, and it has been served by the intended origin, within the validity frame of the delegation. There seems to be an issue with hierarchical caching: the resource can’t be served from a downstream Proxy which is different from the one that was originally delegated unless each Proxy in the delivery chain wraps the received message with its own credentials?

7. References

7.1. Normative References

[I-D.ietf-core-coap]  

7.2. Informative References


[I-D.rahman-core-sleepy-problem-statement]

Authors’ Addresses

Thomas Fossati
KoanLogic

Email: tho@koanlogic.com

Pierpaolo Giacomin
Freelance

Email: yrz@anche.no

Salvatore Loreto
Ericsson
Hirsalantie 11
Jorvas 02420
Finland

Email: salvatore.loreto@ericsson.com