core
Internet-Draft
Intended status: Standards Track
Expires: April 25, 2013

B. Greevenbosch
Huawei Technologies
J. Hoebeke
I. Ishaq
iMinds-IBCN/UGent

October 22, 2012

CoAP Profile Description Format
draft-greevenbosch-core-profile-description-01

Abstract

This document provides a profile description format, that can be used to express capabilities of a CoAP server.

Note

Discussion and suggestions for improvement are requested, and should be sent to core@ietf.org.

Status of this Memo

Copyright Notice

Table of Contents

1. Requirements notation

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].

2. Introduction

The Constrained Application Protocol (CoAP) [I-D.ietf-core-coap] is a RESTful protocol for constrained nodes and networks.

Often, a client first learns about a resource through the link format [I-D.ietf-core-link-format]. The link format only provides basic information, for example the resource URL. However, it would be good if the client could get more extensive information on the resources when required. This document defines a profile description format, which can be used to signal several parameters about resources and their servers.

One of the main features of the CoAP protocol is the ability to include CoAP options. These options can be either elective or critical. A message with an unsupported critical option will be rejected, whereas unsupported elective options will be ignored.

Even though it is well defined how the server must respond to unsupported options, it is useful for the client to know which options are supported in advance. In this way, it can determine which options are viable to use in a transaction, and which features cannot be exploited. This specification allows signalling of the supported options by the resource.

Another important feature of a CoAP server is which content formats it supports. CoAP provides a mechanism for the client to indicate to the server which content format the client prefers. The use of profiles allows signalling what content formats are supported by the server, so that the client can decide which content type it prefers.

It is anticipated that more profile descriptions will become available in the future.

3. Obtaining the profile information

Similar to the link format from [RFC6690], the profile interface uses a well-known resource URI as a pointer to the profile description.

The host component of the URI of the profile description should be equal to the URI of the associated resource, whereas the path component begins with ".well-known" as specified in [RFC5785]. The complete path component equals ".well-known/profile".

For example, if the client wants to get the profile description of a server with URI "www.example.org", it can send a GET request for "coap://www.example.org/.well-known/profile". In this case the server SHOULD respond with the profile details of all resources on the server. The server MAY use the reserved resource name "." to provide a default profile. This default profile applies to all resources that are not specifically listed in the profile (e.g. because they do not have their individual profile) and describes the general CoAP capabilities of the server. If a resource has its own profile, then this profile MUST be used and the default profile field MUST be ignored.

If only the profile of a particular resource is needed, the client SHOULD send a GET request including the "path" URI-query. If the resource has no specific profile the server MUST respond with the default profile.

For example, the profile of the sensor "coap://www.example.org/s" can be acquired with a GET to:
"coap://www.example.org/.well-known/profile?path=s".

Section 5 covers this in more detail.

4. The Profile Format

The profile description is formatted as a JSON document. It consists of several profile fields, each of which has associated parameters.

4.1. The path "path" profile field

The "path" profile field contains the Uri-Path component associated with the resource. It can be used to filter on certain profile properties, as described in Section 5.

4.2. The options "op" profile field

The options "op" profile field contains a list of options that are supported by a resource. It has the format depicted in Figure 1, where op1, op2, ... are integers representing the option numbers of the supported options, as defined in [I-D.ietf-core-coap] and/or through IANA. The option numbers MUST appear in numerical order, starting with the lowest number.

```
"op":[op1,op2,...]
```

Figure 1: "op" syntax

When including the "op" profile field in the profile description of a resource, all option numbers of the CoAP options supported by that resource MUST be included. Options that are not supported by the resource MUST NOT be included in the "op" profile field.

If the "op" profile field is available, the receiving party SHALL assume a non-listed option is not supported by the associated resource.

4.3. The Content-Formats "cf" profile field

The content formats "cf" profile field indicates which content formats are supported. It has the format depicted in Figure 2, where cf1, cf2, ... are integers representing the numbers of the supported content formats, as defined in [I-D.ietf-core-coap] and/or through IANA. The content format numbers MUST appear in numerical order, starting with the lowest number.

```
"cf":[cf1,cf2,...]
```

Figure 2: "cf" syntax

When including the "cf" profile field in the profile description of a resource, all content formats of the CoAP options supported by that

resource MUST be included. Content formats that are not supported by the resource MUST NOT be included in the "cf" profile field.

If the "cf" profile field is available, the receiving party SHALL assume a non-listed content format is not supported by the associated resource.

4.4. The Block-Sizes "b1s" and "b2s" profile fields

The block sizes "b1s" and "b2s" profile fields indicate which block sizes are supported for Block1 and Block2 options when block-wise transfer is used. It has the format depicted in Figure 3, where b1s1, b1s2, ... are three-bit unsigned integers indicating the size of a block to the power of two. Thus block size = 2**(b1 + 4). The allowed values of b1 are 0 to 6, i.e., the minimum block size is 2**(0+4) = 16 and the maximum is 2**(6+4) = 1024. The block-size numbers MUST appear in numerical order, starting with the lowest number (see [I-D.ietf-core-block]).

```
"b1s":[b1s1,b1s2,...]
"b2s":[b2s1,b2s2,...]
```

Figure 3: "b1s" and "b2s" syntax

When including the "b1s" or the "b2s" profile fields in the profile description of a resource, all respective Block1 and Block2 sizes that are supported in block-wise transfer by that resource MUST be included. Block sizes that are not supported by the resource MUST NOT be included in the "b1s" or the "b2s" profile fields.

If the "b1s" or the "b2s" profile fields are available, the receiving party SHALL assume a non-listed block size is not supported by the associated resource. If only one of the "b1s" and the "b2s" profile fields is available, the receiving party SHALL assume that the other block transfer is not supported by the associated resource.

5. Usage of URI Queries

To specify which information is needed, the client MAY include an "Uri-Query" option in its request for the profile description. The server SHOULD understand and process this information, although constraint servers MAY omit the functionality. In the latter case, they SHOULD return the same results as if the "Uri-Query" option was not included.

The URI Queries are of the form "N=V", where N is the name of the field to filter on, and V is the desired value.

For example, if the client wants to know all resources on the server that support content format "application/json", which has the number 50 (see [I-D.ietf-core-coap]), then it will include a "Uri-Query" option with the value "cf=50".

When the request contains multiple "Uri-Query" options, "AND" semantics hold.

6. Forward compatibility

To allow addition of new profile fields in the future, unknown profile fields SHOULD be ignored by the client.

7. Examples

The following is an example of a camera sensor at "coap://www.example.org/cam", that supports the "Uri-Host" (3), "ETag" (4), "Uri-Port" (7), "Uri-Path" (11), "Content-Format" (12), "Token" (19), "Block2" (23) and "Proxy-Uri" (35) options.

The supported content formats are "text/plain" (0), "application/link-format" (40) and "application/json" (50).

The camera can support Block2 with Block sizes of 256 or 512 bytes.

```
Req: GET coap://www.example.org/.well-known/profile

Res: 2.05 Content (application/json)
{
  "profile":
  {
    "path":"cam",
    "op":[3,4,7,11,12,19,23,35],
    "cf":[0,40,50]
    "b2s":[4,5]
  }
}
```

If the server also supports three other resources, such as a temperature sensor (which can do observe), a humidity and a fire detector, the request/response pair would look as follows:

```
Req: GET coap://www.example.org/.well-known/profile

Res: 2.05 Content (application/json)
{
  "profile":[
    {
      "path":".",
      "op":[3,4,7,11,12,19,35],
      "cf":[0]
   },
   {
      "path":"cam",
      "op":[3,4,7,11,12,19,23,35],
      "cf":[0,40,50]
      "b2s":[4,5]
    },
    {
      "path":"temperature",
      "op":[3,4,6,7,11,12,19,35],
      "cf":[0]
    },
  ]
}
```

Please note that the response did not include profiles for the "fire" and "humidity" resources. Instead it included a default profile that applies for these two not explicitly mentioned resources.

If the client now wants to get the resources that support content-format "application/json" (50) it looks as follows:

```
Req: GET coap://www.example.org/.well-known/profile?cf=50

Res: 2.05 Content (application/json)
{
  "profile":
  {
    "path":"cam",
    "op":[3,4,7,11,12,19,23,35],
    "cf":[0,40,50]
    "b2s":[4,5]
  }
}
```

8. Open topics

8.1. Open since v00

- How to signal the client profile?
- Which other profile data needs signalling?
- A natural content format for a camera would be JPEG. Therefore the "image/jpeg" content format may need CoAP registration.
- Currently, support of observe can be signalled in the link format as well as through the mechanism described here.
- Is it needed to signal the profile description on a resource basis, or would it be enough to have only one, associated with the server?
- Fix the order in which the profile fields must appear?
- Make a distinction between "critical" and "elective" profile fields?

8.2. Open since v01

- Addition of the "path" option seems to create overlap with the link format.
- For the time being, text about the hierarchy of profiles in servers, batches and resources has been removed. This leads to a requirement to provide the profile description for each separate resource. A mechanism to re-introduce hierarchy may make significantly reduce the profile description verboseness.

9. Change log

9.1. Changes in v01

- Changed from /p suffix to usage of ".well-known/profile"
- Added support of Uri-Query
- Updated option numbering according to [I-D.ietf-core-coap]
- Changed Media Type and "mt" to Content Format and "cf", in accordance with [I-D.ietf-core-coap]
- Expanded examples
- Removed text about the hierarchy
- Added default profile "."
- Added "b1s" and "b2s" fields for block size

10. Security Considerations

For general CoAP security considerations see [I-D.ietf-core-coap].

In an unprotected environment, an attacker can change the profile description. For example, the list of supported options may be changed. This could cause the client to make a wrong decision on which mechanisms to use. However, such a threat is normal in environments that lack secure authentication.

11. IANA Considerations

- o A registry for profile fields as well as possible values needs to be set up.

- o The ".well-known/profile" path component must be registered.

12. Acknowledgements

The authors would like to thank Kepeng Li for his ideas and feedback.

13. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5785] Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known Uniform Resource Identifiers (URIs)", RFC 5785, April 2010.

[RFC6690] Shelby, Z., "Constrained RESTful Environments (CoRE) Link Format", RFC 6690, August 2012.

[I-D.ietf-core-block] Bormann, C. and Z. Shelby, "Blockwise transfers in CoAP", draft-ietf-core-block-10 (work in progress), October 2012.

[I-D.ietf-core-coap] Shelby, Z., Hartke, K., Bormann, C., and B. Frank, "Constrained Application Protocol (CoAP)", draft-ietf-core-coap-12 (work in progress), October 2012.

[I-D.ietf-core-link-format] Shelby, Z., "CoRE Link Format", draft-ietf-core-link-format-12 (work in progress), May 2012.

Authors' Addresses

Bert Greevenbosch
Huawei Technologies Co., Ltd.
Huawei Industrial Base
Bantian, Longgang District
Shenzhen 518129
P.R. China

Phone: +86-755-28978088
Email: bert.greevenbosch@huawei.com

Jeroen Hoebeke
iMinds-IBCN/UGent
Department of Information Technology
Internet Based Communication Networks and Services (IBCN)
Ghent University - iMinds
Gaston Crommenlaan 8 bus 201
Ghent B-9050
Belgium

Phone: +32-9-3314954
Email: jeroen.hoebeke@intec.ugent.be

Isam Ishaq
iMinds-IBCN/UGent
Department of Information Technology
Internet Based Communication Networks and Services (IBCN)
Ghent University - iMinds
Gaston Crommenlaan 8 bus 201
Ghent B-9050
Belgium

Phone: +32-9-3314946
Email: isam.ishaq@intec.ugent.be