                     Conditioanal observe in CoAP
                  draft-li-core-conditional-observe-00

Abstract

   CoAP is a RESTful application protocol for constrained nodes and
   networks. This document defines a new mechanism in CoAP Observe so
   that a CoAP client can conditional observe a resource on 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. **Introduction**

   CoAP [I-D.ietf-core-coap] is an Application Protocol for Constrained
   Nodes/Networks.The observe [I-D.ietf-core-observe] specification
   describes a protocol design so that a CoAP client and server can
   using the subject/observer design pattern to establish the
   observation relationship. When observe used, the CoAP client can get
   the notification response whenever the state changed of the observed
   resource, however, in some scenarios, the CoAP client may only care
   parts of the state change of the resource, other state change might
   be meaningless. This inability to suppress additional notification
   results in superfluous traffic. This memo defines a new CoAP option
   "Condition" that can be used to allow the CoAP client to condition
   the observe request, and only when such condition met, the CoAP
   server shall send the notification response with the latest state
   change. When such a condition fails, the CoAP server does not need
   to send the notification response.

2. **overview**

   A GET request that includes an Observe Option creates an observation
   relationship. When a server receives such a request, it firt
   services the request like a GET request without this option and, if
   the resulting response indicates success, establishes an observation
   relationship between the client and the target resource. The client
   is notified of resource state changes by additional responses sent in
   reply to the GET request to the client.

   CoAP is used for Constrained Networks, especially used for
   transporting sensor data. But Different sensor equipments have
   different properties, e.g. different data unit, different response
   time. When a client wants to collect information from a sensor, it
   does not want to receive useless information, it hopes that the senor
   only responses with what the client wants.

   Let's see an example here,

```
     CLIENT                                                    SERVER
       |                                                         |
       | GET/temperature observe:0                    ------>    |
       |                                                         |
       | <------    2.05 Content observe:5 Payload:22            |
       |                                                         |
       |                                                         |
       |                                                         |
       | <------    2.05 Content observe:10 Payload:22.3         |
       |                                                         |
       |                                                         |
       |                                                         |
       | <------    2.05 Content observe:15 Payload:22.6         |
       |                                                         |
```
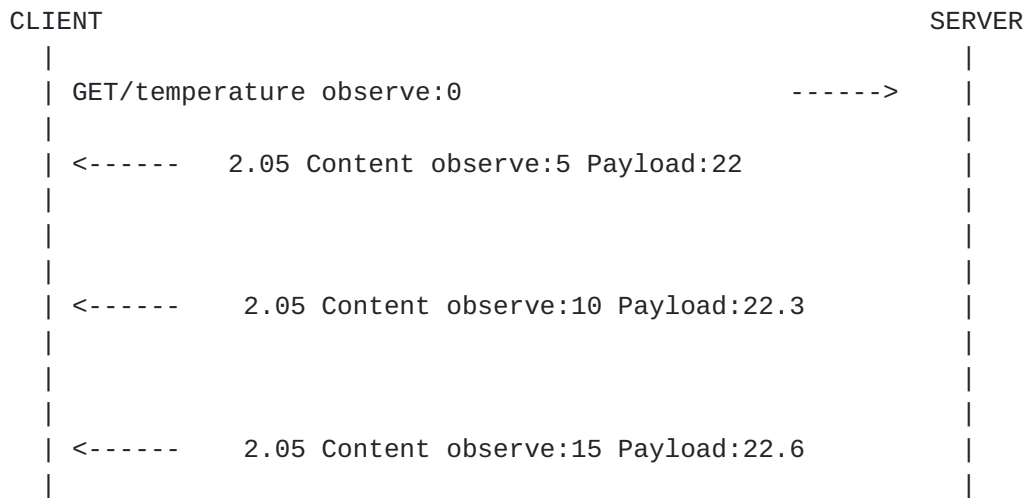
                        Figure 1: GET request with observe

   In this example, the sensor worked as a server, and it collect the
   resource data every 5 seconds, when the client observe a resource on
   the server, it will receive the response whenever the server update
   the resource, that is to say, mostly every 5 seconds the clients will
   receive a notification response. However, the client might be a
   quite simple equipment not too sensitive to the resource state
   change, so it may not want to receive the notification such often.
   One possible solution we can do is to change the sensor's parameter,
   for example, shorten the collecting frequency, but the sensor could
   provide services for many other clients, it is hard to fine the best
   configuration so it can fit all the requirements.

2.1. Terminology

   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].

3. Requirements

   As a summary, here is the required functionality to solve the
   presented issues:

   REQ 1: It must be possible to include the client's requirements into
   an observe request

4. Overview of Operation

   Whenever a client initiates an Observation relationship, it sends a

GET request with a Condition Option (see section 5 for the option
definition). The Condition option includes the condition required by
the client, it can be the desired minimum response time, or the
minimum step change(see section 6 for the definition of condition
type) . When a server receive such a request, it first services the
request the same way as described in observe
draft[I-D.ietf-core-observe] that established an observation
relationship between the client and the target resource.Then if the
server supports the Condition option, it needs to analyze the
Condition option to find the requirements by the client.

For example,

condition: 1/0/10

Whenever the resource state changed on the server, it needs to check
the condition, only when it meet, the server shall send the
notification response to the client, otherwise, the server does not
need to send any response to the client.

5. Definition of Condition Option

```
+------+-----+-----------+-----------+--------+---------------+
| Type | C/E | Name      | Data type | Length | Default       |
+------+-----+-----------+-----------+--------+---------------+
|  22  |  E  | Condition | uint      | 1-3 B  |               |
+------+-----+-----------+-----------+--------+---------------+
              table 1: Condition Option number
```

6. Using the Condition option

The Condition option is used to indicate the condition requirement
requested by a CoAP client. It must be used together with Observe
option.

The condition Option, when present together with the observe request,
it represent the condition required by the client. The server needs
to follow the general procedure as described in observe
draft[I-D.ietf-core-observe] but take the Condition option into
account.

Since the condition Option is elective, an observe request that
includes the Condition Option will automatically fall back to a basic
observe request if the server does not support the Condition
Option.There is no default value for the Condition option.

The condition type defined in this documents are as follows:

The size of the condition option is not fixed by the protocol.The value of the defined condition type can be ranged from 0 to 2^4 (16), and from 0 to 2^12(4096), and from 0 to 2^20(1048576), it all depends on the implementation to choose.

The Condition option may occur more than once, when multiple Condition options present in an observe GET request, it means that the initiator has multiple condition requirements, and the first condition option shows in the request has the highist proiority.

```
  0
  0 1 2 3 4 5 6 7 8 9
 +-+-+-+-+-+-+-+-+-+-+
 | TYPE  | M | VAL   |
 +-+-+-+-+-+-+-+-+-+-+

  0                   1
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 | TYPE  | M |           VAL         |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

  0                   1                   2
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 | TYPE  | M |           VAL                         |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
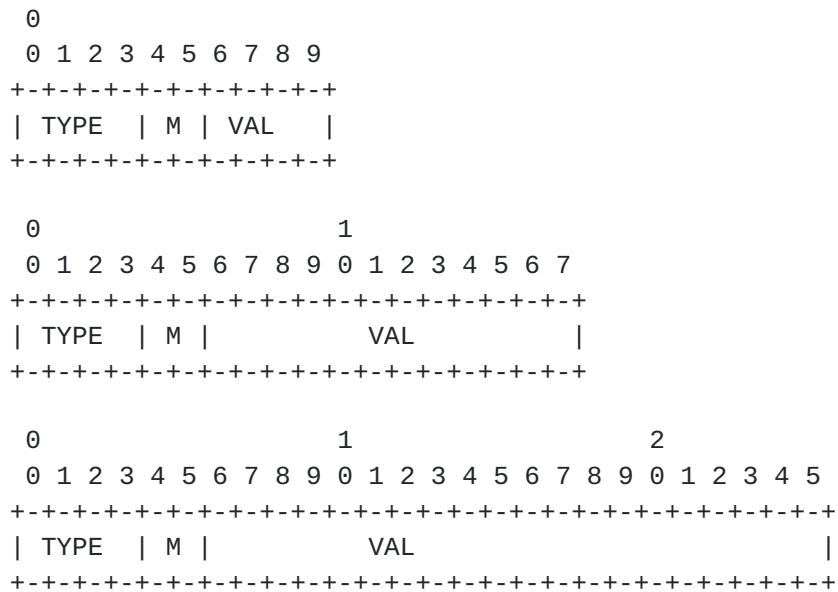
               table 2: Condition Option value

TYPE: condition type. the condition type is a 4 bit integer indicate the type of the condition that used in the observe request. Each number of TYPE represent the type ID of a specific condition type, for example, "1" represents the minium response time, "2" represent the step.Blow has the further definition of the condition type.

M: method. the method is a 2 bit intrger indicate the method used in the condition. Blow has the further definition of the method attribute.

VAL: condition value. is a variable-size(4,12, or 20 bit) unsigned integer indicate the value of the condition.

```
        +----------------------+-----+
        |Condition type        | Id. |
        +----------------------+-----+
        |  minium response time|  1  |
        +----------------------+-----+
        |  step                |  2  |
        +----------------------+-----+
        |  range               |  3  |
        +----------------------+-----+
```
            table 3: Condition Option type

minium response time: When present, it indicates that the condition
required by the initiator is the minium response time. The value of
the minium response time is set in the payload of the request.This
condition indicates the minimum time the server should wait between
sending notification responses.

step: When present, it indicates that the condition required by the
initiator is the change step. Its value is set in the payload of the
request.This condition indicates the minium state change of a
resource before the server can send a new notification response.

range: When present, it indicates that the condition required by the
initiator is the state change range. This condition indicates that
only when the state value is under the range of this condition, the
server shall send a new notification response. The value and the
method attributes decide the range of the condition, e.g, set the
method to ">", and value to "20", means that the range is bigger than
20, so only the state value bigger than 20, that the server shall
send a new notification response to the client.

If multiple conditions with the same condition type present in a
request, the priority is the same, and the relationship is "AND".

```
        +----------------------+-----+
        |Method                | Id. |
        +----------------------+-----+
        |   =                  | 0   |
        +----------------------+-----+
        |   >                  | 1   |
        +----------------------+-----+
        |   <                  | 2   |
        +----------------------+-----+
```
            table 4: Condition method

7. Examples

This section gives a number of short examples with message flows to
illustrate the use of Condition option in a observe GET request.

The first example (Figure 2) shows that the client set the Conditon
option to 1/0/10, it means that the server shill wait at least 10s
between sending notification responses to the client .

```
   CLIENT                                               SERVER
     |                                                    |
     |      GET/temperature, observe:0,Condition:1/0/10----->    |0s
     |                                                    |
     |                                                    |
     | <------ 2.05Content,observe:5,payload:22                22|5s
     |                                                    |
     |                                                  22.4|10s
     |                                                    |
     | <------ 2.05Content,observe:15,payload:22.5            22.5|15s
     |                                                    |
     |                                                   23  |20s
     |                                                    |
     |   <------ 2.05Content,observe:25,payload:22.8          22.8|25s
     |                                                    |
```

Figure 2: Condition Option with value 1/0/10

The second example (Figure 3) shows the client set the Conditon
option value to 2/0/1, it means that the server will send the
notification response to the client if the resource state change
bigger than 1.

```
   CLIENT                                               SERVER
     |                                                    |
     | GET/temperature, observe:0,Condition:2/0/1      ----->  |
     |                                                    |
     |                                                    |
     | <------ 2.05Content,observe:5,payload:22               |22
     |                                                    |
     |                                                  |22.5
     |                                                    |
     | <------ 2.05Content,observe:20,payload:23.2           |23.2
     |                                                    |
     |                                                  |23.6
     |                                                    |
     |   <------ 2.05Content,observe:35,payload:24.3         |24.3
```
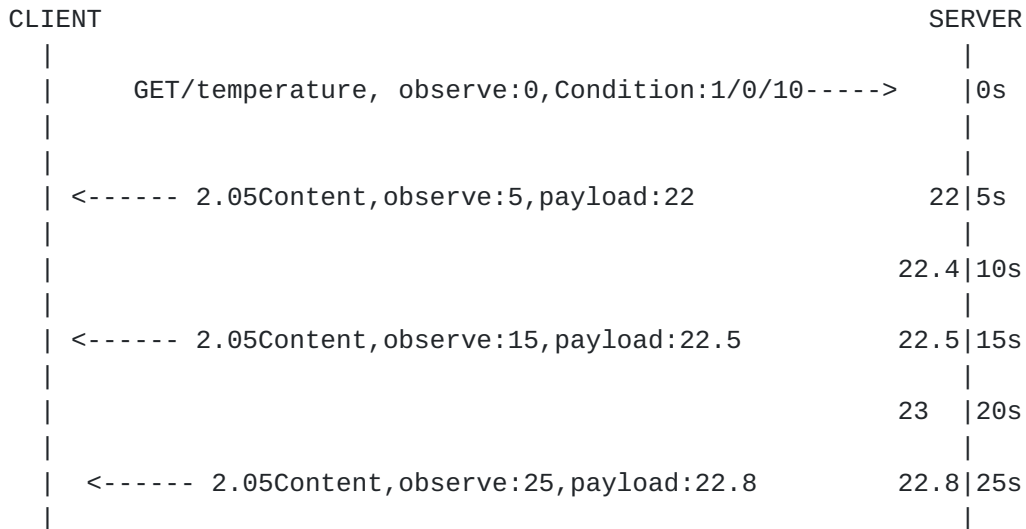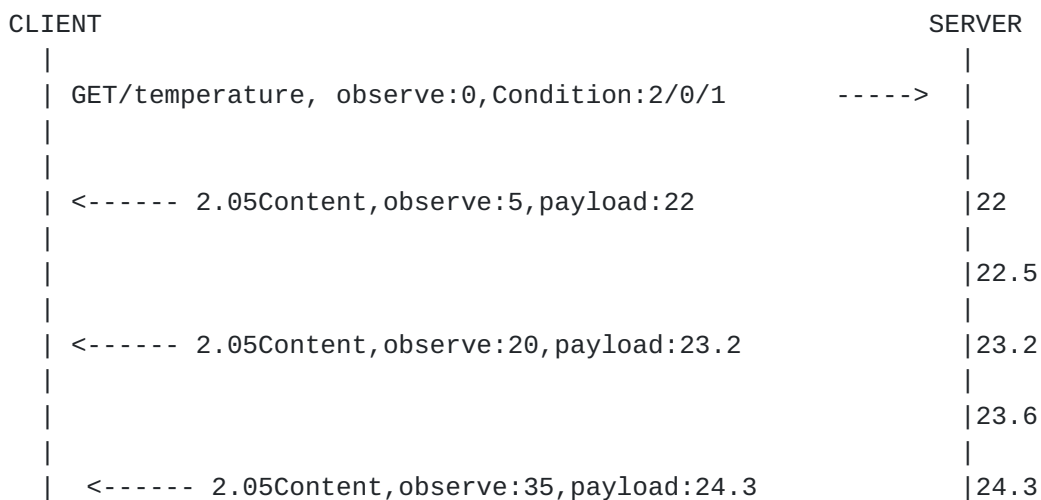
Figure 3: Condition Option with value 2/0/1

The third example (Figure 4) shows the client set the Conditon option
value to 3/1/5, it means that the server will send the notification
response to the client only if the resource value bigger than 5.

```
     CLIENT                                              SERVER
       |                                                  |
       | GET/temperature, observe:0,Condition:3/1/5    -----> |
       |                                                  |
       |                                                  |
       | <------ 2.05Content,observe:5,payload:4          |4
       |                                                  |
       |                                                  |3
       |                                                  |
       | <------ 2.05Content,observe:25,payload:6         |6
       |                                                  |
       |                                                  |4.5
       |                                                  |
       |   <------ 2.05Content,observe:25,payload:7       |7
```

Figure 4: Condition Option with value 3/1/5

The fourth example (Figure 5) shows the client adds two range
Conditon options in the request , one sets to 3/1/5, another one sets
to 3/2/15 it means that the range is within 5 and 15.

```
     CLIENT                                              SERVER
       |                                                  |
       | GET/temperature, observe:0,Condition:3/1/5,      |
       |                         Condition:3/2/15    ----->|
       |                                                  |
       |                                                  |
       | <------ 2.05Content,observe:5,payload:4          |4
       |                                                  |
       |                                                  |3
       |                                                  |
       | <------ 2.05Content,observe:25,payload:12        |12
       |                                                  |
       |                                                  |16
       |                                                  |
       |   <------ 2.05Content,observe:25,payload:14      |14
```

Figure 5: Two Condition Option with the same condition type

8. Security Considerations

   As the Condition option is used together with the Observe option,
   when it used it must follow the secutity considertions as described
   in Observe draft[I-D.ietf-core-observe].

9. IANA Considerations

9.1. Condition option registry

   This draft adds the following option number to the CoAP Option
   Numbers registry of [I-D.ietf-core-coap]

```
            +--------+--------------+----------------+
            | Number | Name         | Reference      |
            +--------+--------------+----------------+
            |  22    | Condition    | [RFCXXXX]      |
            +--------+--------------+----------------+
              table 5: Condition Option number
```

9.2. Condition type registry

   The Condition types defined in this draft are identifed by a string,
   such as"step". In order to minimze the overhead of using these
   condition type, this document defines a registry for the condition
   types to be used in CoAP and assigns each a numeric identifier.

   Each entry in the registry must include the condition type registered
   with IANA, the numeric identifier in the range 0-15 to be used for
   that condition type in CoAP, and a reference to a document defining
   the usage of that condition type.

   Initial entries in this registry are as follows:

```
            +-----------------------+-----+-------------+
            |Condition type         | Id. | Reference   |
            +-----------------------+-----+-------------+
            |  minium response time |  1  |[RFCXXXX]    |
            +-----------------------+-----+-------------+
            |  step                 |  2  |[RFCXXXX]    |
            +-----------------------+-----+-------------+
            |  range                |  3  |[RFCXXXX]    |
            +-----------------------+-----+-------------+
                    table 6: Condition Option type
```

9.3. Condition method registry

   The condition method used in this draft are as follow:

```
        +----------------------+-----+------------+
        |Method                | Id. | Reference  |
        +----------------------+-----+------------+
        |    =                 | 0   |[RFCXXXX]   |
        +----------------------+-----+------------+
        | >                    |  1  |[RFCXXXX]   |
        +----------------------+-----+------------+
        |  <                   |  2  |[RFCXXXX]   |
        +----------------------+-----+------------+
          table 7: Condition method
```

10. Further Considerations

11. Acknowledgements

12. Normative References

   [I-D.ietf-core-coap]
             Shelby, Z., Hartke, K., Bormann, C., and B. Frank,
             "Constrained Application Protocol (CoAP)",
             draft-ietf-core-coap-08 (work in progress), October 2011.

   [I-D.ietf-core-link-format]
             Shelby, Z., "CoRE Link Format",
             draft-ietf-core-link-format-11 (work in progress),
             January 2012.

   [I-D.ietf-core-observe]
             Hartke, K., "Observing Resources in CoAP",
             draft-ietf-core-observe-04 (work in progress),
             February 2012.

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

Author's Address

    Shitao Li
    Huawei Technologies
    Huawei Base
    101 Software Avenue, Yuhua District
    Nanjing, Jiangsu 210012
    China

    Phone: +86-25-56624157
    Email: lishitao@huawei.com