Group Communication for the Constrained Application Protocol (CoAP)

draft-ietf-core-groupcomm-bis-06

Esko Dijk, IoTconsultancy.nl
Chonggang Wang, InterDigital
Marco Tiloca, RISE

IETF 113 Meeting – Vienna – March 25, 2022
Following IETF 112 & CoRE interim...

Examples were added for:
- Encoding application group names in CoAP requests (issue #28)
- Discovery of CoAP groups and application groups on CoAP servers (issue #29)
- Message exchanges for group communication – new Appendix B

Removed the (confusing) usage of URI Host Option to encode an application group name

Text & editorial improvements

All open issues now closed!
2.2.1 - Name encoding of app groups

- Application group name 'gp1' encoded in URI path

- Examples updated

(Further examples not shown)
2.2.3.2 – Group discovery examples

› Discover groups without Resource Directory – using CoAP Discovery

› Given a CoAP group, discover …
  – The associated application groups
  – The servers in it and each group’s resources

(Further examples not shown)
### Appendix B – Message exchange examples

#### Example 1 (basic)
- Request over multicast
- 3 Responses follow
- 1 Response is lost
- With a **special twist** (allowed for server)

<table>
<thead>
<tr>
<th>Client</th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>+-------+------&gt;</td>
<td></td>
<td></td>
<td>Source: ADDR_CLIENT:PORT_CLIENT</td>
</tr>
<tr>
<td>\</td>
<td></td>
<td>Destination: ADDR_GRP:PORT_GRP</td>
<td></td>
</tr>
<tr>
<td>,-------&gt;</td>
<td></td>
<td></td>
<td>Header: GET (T=NON, Code=0.01, MID=0x7d41)</td>
</tr>
<tr>
<td>\</td>
<td></td>
<td></td>
<td>Token: 0x86</td>
</tr>
<tr>
<td>\</td>
<td></td>
<td></td>
<td>Uri-Path: &quot;gp&quot;</td>
</tr>
<tr>
<td>\</td>
<td></td>
<td></td>
<td>Uri-Path: &quot;gp1&quot;</td>
</tr>
<tr>
<td>\</td>
<td></td>
<td></td>
<td>Uri-Path: &quot;temperature&quot;</td>
</tr>
<tr>
<td>&lt;---------------+</td>
<td></td>
<td></td>
<td>Source: ADDR_A:PORT_GRP</td>
</tr>
<tr>
<td>2.05</td>
<td></td>
<td></td>
<td>Destination: ADDR_CLIENT:PORT_CLIENT</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Header: 2.05 (T=NON, Code=2.05, MID=0x60b1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Token: 0x86</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Payload: &quot;22.3 C&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>X---------------+ Source: ADDR_B:PORT_GRP</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Destination: ADDR_CLIENT:PORT_CLIENT</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Header: 2.05 (T=NON, Code=2.05, MID=0x01a0)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Token: 0x86</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Payload: &quot;20.9 C&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;---------------------+ Source: ADDR_C:PORT_C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Destination: ADDR_CLIENT:PORT_CLIENT</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Header: 2.05 (T=NON, Code=2.05, MID=0x952a)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Token: 0x86</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Payload: &quot;21.0 C&quot;</td>
</tr>
</tbody>
</table>
Appendix B – Message exchange examples

Example 3 (advanced)
- Request over multicast
- 3 Responses follow
- Block-wise transfer used
Appendix B – Message exchange examples

Example 3 (continued)

- Client continues with unicast block-wise transfer for each server

Client | GET | B | C

Source: ADDR_CLIENT:PORT_CLIENT
Destination: ADDR_A:PORT_A
Header: GET (T=CON, Code=0.01, MID=0x7d42)
Token: 0xa6
Uri-Path: "gp"
Uri-Path: "gp1"
Uri-Path: "log"
Block2: 1/0/64

2.05

Source: ADDR_A:PORT_A
Destination: ADDR_CLIENT:PORT_CLIENT
Header: 2.05 (T=ACK, Code=2.05, MID=0x7d42)
Token: 0xa6
Block2: 1/1/64
Payload: 0x0a01 ...

GET

Source: ADDR_CLIENT:PORT_CLIENT
Destination: ADDR_A:PORT_A
Header: GET (T=CON, Code=0.01, MID=0x7d43)
Token: 0xa7
Uri-Path: "gp"
Uri-Path: "gp1"
Uri-Path: "log"
Block2: 2/0/64

2.05

Source: ADDR_A:PORT_A
Destination: ADDR_CLIENT:PORT_CLIENT
Header: 2.05 (T=ACK, Code=2.05, MID=0x7d43)
Token: 0xa7
Block2: 2/0/64
Payload: 0x0a02 ...
Example 3 (continued)

- Client continues with unicast block-wise transfer for each server.
Next steps

› Ready for Working Group Last Call (for -06)
Thank you!

Comments/questions?

https://github.com/core-wg/groupcomm-bis/
Goal

› Normative successor of experimental RFC 7390
  – Obsoletes RFC 7390, Updates RFC 7252 / 7641

› New standard reference for implementations now based on RFC 7390

› Scope
  – CoAP group communication, including latest features: Observe/Blockwise/Security …
  – Unsecured & group-OSCORE-secured
  – Definition of group types & Secure group configuration
Motivation (backup slide)

› RFC 7390 was published in 2014
  – CoAP functionalities available by then were covered
  – No group security solution was available to indicate
  – It is an Experimental document (started as Informational)

› What has changed?
  – More CoAP functionalities have been developed (Block-Wise, Observe)
  – RESTful interface for membership configuration is not really used
  – Group OSCORE provides group end-to-end security for CoAP

› Practical considerations
  – Group OSCORE clearly builds on RFC 7390 normatively
  – However, it can refer RFC 7390 only informationally