Resource-constrained Industrial Things – Proposal for the Adaptation of CoAP to EtherNet/IP™

Jonas Green, Björn Otterdahl
HMS Industrial Networks AB

February 22, 2017
Why EtherNet/IP over CoAP?

- Connect all devices in a factory to EtherNet/IP
  - Even small and simple sensors and actuators
  - No need for routers
  - Enable more diagnostic data from the devices

- Make EtherNet/IP cheaper and more simple to implement
  - Remove TCP and the Encapsulation protocol and replace with CoAP
  - Less computing power and less memory required by the sensor system

- Make the devices IoT Ready in a secure way
  - Leverage on home automation to enable cloud connectivity for every sensor
  - Use the security features used by the home automation business
• Continuation on the work on Resource constrained EtherNet/IP device
  • Presented 2015 by Paul Brooks and Dayin Xu

• Quick recap:
  • Widen the ODVA ecosystem with IIoT solutions
  • Cost and size reduction needed
  • EtherNet/IP over UDP reduces size/cost
  • Single Pair Ethernet reduces size/cost
  • CIP transport barrier – relies on TCP
Why do we need a resource constrained design?

- Introduce sensors and small actuators to the ODVA ecosystem via Ethernet connectivity
- Enable data and diagnostics to achieve smart operations from every thing on the factory floor
- Cheaper and smaller solution is necessary
Introduction

• Bachelor thesis in collaboration with Halmstad University 2016
  • Evaluated different IoT protocols to solve reliability for UDP based EtherNet/IP communication. (*CoAP, AMQP, MQTT and DDS were compared*)
  • CoAP is request/response oriented
  • Demo implementation
Constrained Application Protocol

• Modern IoT protocol targeting IoT applications

• Use cases
  – OMA Lightweight M2M – Device management protocol
  – Supported by ARM mbed Device Server
  – Application-layer option to Thread
Constrained Application Protocol

- Machine-2-machine protocol
- Targeting constrained devices

- Low overhead and complexity
  - Built upon UDP
  - 4-byte header

- Lightweight reliability
  - Re-transmissions
  - Duplicate message detection

- Can be secured by DTLS
CoAP header

- Version
- Message type
  - Confirmable, Non-confirmable, Acknowledgement, Reset
- Token length
- Message code
  - Request, Success response, Client error or Server error response
- Message ID
  - Re-transmissions and duplicate message detection

<table>
<thead>
<tr>
<th>Byte</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
<td>0 1 2 3 4 5 6 7</td>
<td>0 1 2 3 4 5 6 7</td>
<td>0 1 2 3 4 5 6 7</td>
<td>0 1 2 3 4 5 6 7</td>
</tr>
<tr>
<td>Fields</td>
<td>Ver</td>
<td>T</td>
<td>TKL</td>
<td>Code</td>
</tr>
</tbody>
</table>
CoAP options & payload

- **Options**
  - Present in both requests and responses
  - Type-Length-Value format
  - Multiple instances in single message
  - 15 options specified

- **Payload**
  - Remaining part of the datagram size
  - Format determined by the Content-Format option

<table>
<thead>
<tr>
<th>Uri Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Content-Format</td>
<td>Indicates the representation format of the message payload</td>
</tr>
<tr>
<td>Uri-Host</td>
<td>Specifies the Internet host</td>
</tr>
<tr>
<td>Uri-Port</td>
<td>Specifies the transport-layer port</td>
</tr>
<tr>
<td>Uri-Path</td>
<td>Each instance specifies one segment of the absolute path to the resource to access</td>
</tr>
<tr>
<td>Uri-Query</td>
<td>Each instance specifies one argument parameterizing the resource</td>
</tr>
</tbody>
</table>
CIP adaptation on CoAP

- Replace the Encapsulation layer

<table>
<thead>
<tr>
<th>EtherNet/IP</th>
<th>EtherNet/IP over CoAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>CIP</td>
<td>CIP</td>
</tr>
<tr>
<td>Encapsulation protocol</td>
<td>CoAP</td>
</tr>
<tr>
<td>TCP/UDP</td>
<td>UDP</td>
</tr>
<tr>
<td>Internet Protocol (v4)</td>
<td>Internet Protocol (v4/v6)</td>
</tr>
<tr>
<td>Ethernet</td>
<td>Ethernet, 6LoWPAN etc.</td>
</tr>
</tbody>
</table>
CIP adaptation on CoAP – Messaging

- Address resources with URIs
  - Objects, instances, attributes

- Translate CIP services to CoAP methods
  - Subset of all services
  - Enough to control a device?
  - CoAP option for CIP services

- How to handle status codes?
  - Translate CIP status codes to CoAP response codes
  - Add CoAP options for general and additional status codes

CoAP method | CIP service
--- | ---
GET | Get_Attribute_All / Get_Attribute_Single
POST | Create
PUT | Set_Attribute_All / Set_Attribute_Single
DELETE | Delete
CIP adaptation on CoAP – IO data

• CoAP is by design a messaging protocol
  – CoAP does not offer cyclic data exchange
  – Shall a resource constrained device handle cyclic data exchange?

• Let clients register for data updates
  – Observe resources
  – Publish/subscribe to topics linked to resources
CIP adaptation on CoAP – IO data

• Observing Resources in the Constrained Application Protocol
  – Extension to the CoAP specification (RFC7641)

• Using the CoAP GET request but adds a Observe option

• Clients register to observe a resource
  – In GET request the Observe option tells the server to register an observer
  – Register and deregister is supported

• Servers send notifications when resources are updated
  – Subsequent responses to previous request
  – Sequence number for reordering detection
CIP adaptation on CoAP – IO data

- **CoAP publish/subscribe**
  - Register topics
  - Publish data on topics
  - Subscribe to topics to get data updates (change-of-state)

- Separate broker device

- ”Brokerless” setup
  - Pre-configured topics in the devices
CIP adaptation on CoAP – Security

• CoAP has built in support for DTLS
  – URI: coaps://
  – Port 5684
• For key-exchange CoAP has support for
  – Pre-shared keys
  – X.509 certificates
  – Raw public key
• Uses AES128 for encryption
• Adds an extra burden on the host system
  – Cipher suites must be carefully chosen, Standardized on Elliptic-Curve DH (secp256r1)
  – Hardware acceleration
  – Use raw public keys instead of X.509 certificates (RFC7250)
Example of network traffic

- Get_Attr_Single TCP/IP Interface object, instance 1, attribute 6 (host name)

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Delta</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>2.664916</td>
<td>10.11.20.181</td>
<td>10.11.20.185</td>
<td>CoAP</td>
<td>62</td>
<td>CON, MID:40664, GET, TKN:01 02 03 04, /cip/245/1/6</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>2.667306</td>
<td>0.002390 10.11.20.185</td>
<td>10.11.20.181</td>
<td>CoAP</td>
<td>69</td>
<td>ACK, MID:40664, 2.05 Content, TKN:01 02 03 04</td>
</tr>
</tbody>
</table>
Example of network traffic

- Get_Attr_Single TCP/IP Interface object, instance 1, attribute 6 (host name)

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Delta</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3 2.664916</td>
<td>1.915128</td>
<td>10.11.20.181</td>
<td>10.11.20.185</td>
<td>CoAP</td>
<td>62</td>
<td>CON, MID:40664, GET, TKN:01 02 03 04, /cip/245/1/6</td>
</tr>
<tr>
<td>4</td>
<td>4 2.667306</td>
<td>0.002390</td>
<td>10.11.20.185</td>
<td>10.11.20.185</td>
<td>CoAP</td>
<td>69</td>
<td>ACK, MID:40664, 2.05 Content, TKN:01 02 03 04</td>
</tr>
</tbody>
</table>

Frame 3: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface 0
- Ethernet II, Src: GoodWayI_14:ac:3b (00:50:67:14:ac:3b), Dst: HmsIndus_0f:8c:12 (00:30:11:0f:8c:12)
- Internet Protocol Version 4, Src: 10.11.20.181, Dst: 10.11.20.185
- User Datagram Protocol, Src Port: 52690, Dst Port: 5683
- Constrained Application Protocol, Confirmable, GET, MID:40664

01. .. = Version: 1
..00 .... = Type: Confirmable (0)
.... 0100 = Token Length: 4
Code: GET (1)
Message ID: 40664
Token: 01020304
- Opt Name: #1: Uri-Path: cip
- Opt Name: #2: Uri-Path: 245
- Opt Name: #3: Uri-Path: 1
- Opt Name: #4: Uri-Path: 6
[Response In: 4]
Example of network traffic

- Get_Attr_Single TCP/IP Interface object, instance 1, attribute 6 (host name)

<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Delta</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Length</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3.2664916</td>
<td>1.915128</td>
<td>10.11.20.181</td>
<td>10.11.20.185</td>
<td>CoAP</td>
<td>62</td>
<td>CON, MID:40664, GET, TKN:01 02 03 04, /cip/245/1/6</td>
</tr>
<tr>
<td>4</td>
<td>3.2667306</td>
<td>0.002390</td>
<td>10.11.20.185</td>
<td>10.11.20.181</td>
<td>CoAP</td>
<td>69</td>
<td>ACK, MID:40664, 2.05 Content, TKN:01 02 03 04</td>
</tr>
</tbody>
</table>

- Frame 4: 69 bytes on wire (552 bits), 69 bytes captured (552 bits) on interface 0
- Ethernet II, Src: HmsIndus_0f:8c:12 (00:30:11:0f:8c:12), Dst: GoodWayI_14:ac:3b (00:50:b6:14:ac:3b)
- Internet Protocol Version 4, Src: 10.11.20.185, Dst: 10.11.20.181
- User Datagram Protocol, Src Port: 5683, Dst Port: 52690

**Constrained Application Protocol, Acknowledgement, 2.05 Content, MID:40664**

```
01. ... = Version: 1
.10 ... = Type: Acknowledgement (2)
.... 0100 = Token Length: 4
Code: 2.05 Content (69)
Message ID: 40664
Token: 01020304

Opt Name: #1: Content-Format: application/octet-stream
[Expert Info (Warning/Malformed): Invalid Option Number 65000]
Opt Name: #2: Unknown Option: 00
End of options marker: 255
[Request In: 3]
[Response Time: 0.002390000 seconds]

Payload: Payload Content-Format: application/octet-stream, Length: 12
  Payload Desc: application/octet-stream
```
How to connect to the network

- CoAP Client
- EtherNet/IP Scanner
- Ethernet switch
- EtherNet/IP adapter
- EtherNet/IP adapter
- EtherNet/IP adapter
- EtherNet/IP to CoAP Gateway
- EtherNet/IP over CoAP adapter

Interconnections:
- Standard EtherNet/IP
- CIP over CoAP
- Standard CoAP
How to connect to the network

- EtherNet/IP over CoAp adapter
- Wireless AP
- EtherNet/IP Scanner with support for EtherNet/IP over CoAP
- Ethernet switch with single pair Ethernet
- CoAP Client
- Edge Gateway/router

- Standard EtherNet/IP
- EtherNet/IP over CoAP
- EtherNet/IP over CoAP on Single pair Ethernet
- Wireless link (ex. wifi, 6LowPAN, Bluetooth LE)
- Standard CoAP
Not a finished concept

• CoAP works very well with explicit messaging of CIP
  – Examine how to support all CIP services and CIP Status codes

• No support for exchange cyclic data
  – Examine proposed techniques to replace cyclic data connections
  – How to map these techniques to the Connection Manager and Connection object?

• Investigate how to handle device commissioning
  – Discovery of new devices
Summary

• Low resource protocols for IoT devices exists

• Take advantage of already developed IoT protocols
  – Use as is or use ideas of it when adapting EtherNet/IP

• Possible to use CoAP
  – Support for explicit messaging
  – Cyclic data exchange needs a new approach

• Security of CoAP is well investigated and conforms well with CIP Security

• More to investigate for a final solution
Demo of EtherNet/IP CoAP network adapter

- HMS Anybus CompactCom 40 EtherNet/IP™ adapter
- Open source CoAP stack “libcoap”
- PC with CoAP browser used as originator
• **Next step:** New work in collaboration with Halmstad University, moving technology to a wireless mesh network (6LoWPAN)
Questions and discussion
THANK YOU