



MICROEJ[®]

LwM2M over MQTT

Eclipse IOT Days – March 9, 2017 – Grenoble

frederic.riviere@microej.com



DISCLAIMER

All rights reserved. Information, technical data and tutorials contained in this document are proprietary under copyright Law of Industrial Smart Software Technology (IS2T S.A.) operating under the brand name MicroEJ®. Without written permission from IS2T S.A., copying or sending parts of the document or the entire document by any means to third parties is not permitted. Granted authorizations for using parts of the document or the entire document do not mean IS2T S.A. gives public full access rights.

The information contained herein is not warranted to be error-free.

IS2T® and MicroEJ® and all relative logos are trademarks or registered trademarks of IS2T S.A. in France and other Countries.

Java™ is Sun Microsystems' trademark for a technology for developing application software and deploying it in cross-platform, networked environments. When it is used in this site without adding the “™” symbol, it includes implementations of the technology by companies other than Sun. Java™, all Java-based marks and all related logos are trademarks or registered trademarks of Sun Microsystems Inc, in the United States and other Countries.

Other trademarks are proprietary of their respective owners.

QUICK OVERVIEW

- CoAP - Constrained Application Protocol
 - REST based (GET/POST/PUT/DELETE), Device is viewed as a server
 - Eclipse Reference Implementation: Californium
- MQTT – Message Queue Telemetry Transport
 - Publish/Subscribe paradigm with quality of services
 - Centralized Server (broker)
 - Eclipse Reference implementation: Paho
- LwM2M - Lightweight Machine to Machine
 - Provisioning & Device Management
 - Standard for Core Objects (Device, Firmware Update)
 - Eclipse IOT Reference implementation: Leshan

WHY THIS SPECIFICATION ?

- LwM2M is becoming the standard protocol for device management
- LwM2M specification is tightly coupled with CoAP/UDP/DTLS:

[...] The LWM2M Enabler uses the Constrained Application Protocol (CoAP) with UDP and SMS bindings. Datagram Transport Layer Security (DTLS) provides security for UDP transport layer. [...]

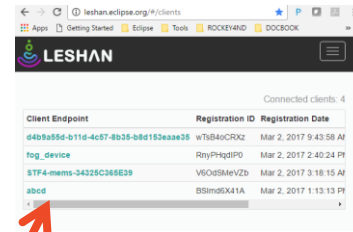
- Still an industrial issue to access a device behind NAT (home topology)
 - DTLS (re-)negotiation latency
 - Keep NAT alive (1 minute ?)

WHY THIS SPECIFICATION ?

- When MQTT is already a project requirement
- MQTT widely used for domain data transfer
 - Device <-> Cloud
 - Optional Peripherals (smartphones, ...)
- A large offering of scalable MQTT Servers for production
- Regular TCP/TLS connection

TOPOLOGY (ECLIPSE PROJECTS)

Fleet Management
Dashboards, ...



HTTPS

LWM2M Server



MQTTS

MQTT Server

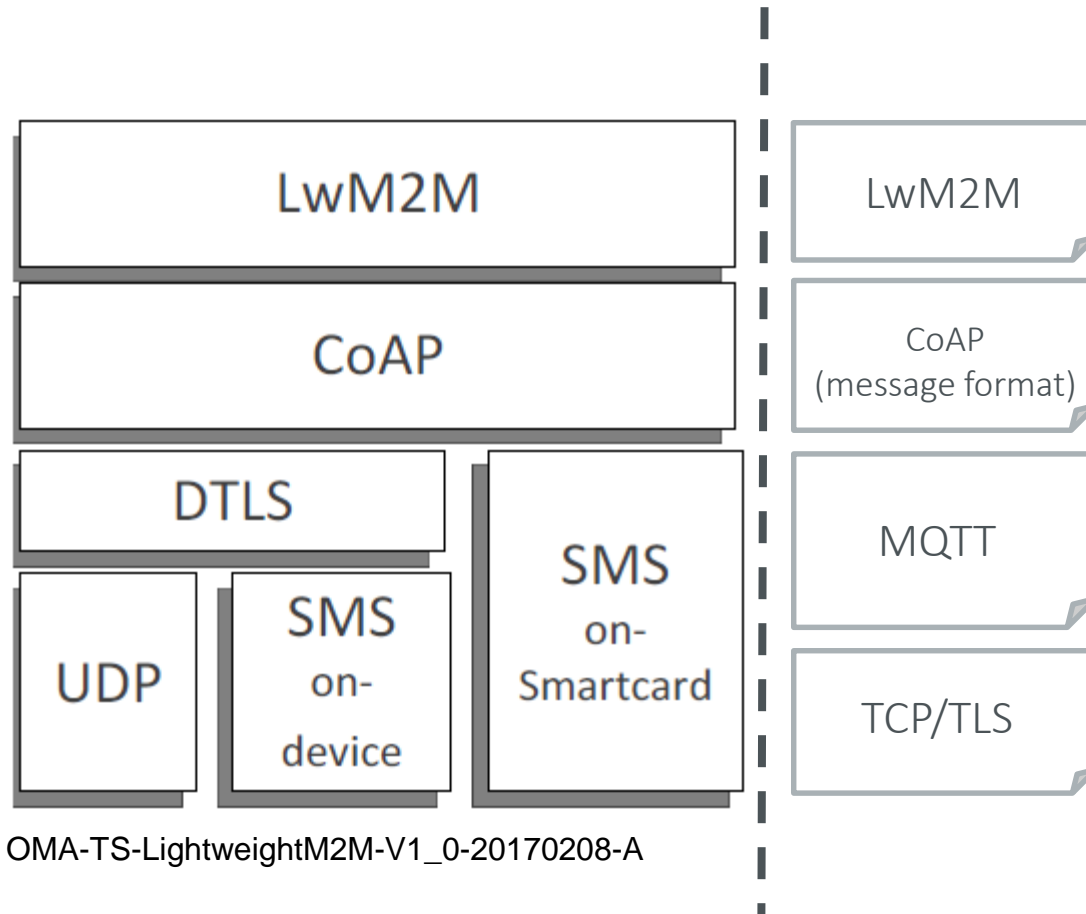


MQTTS

Devices Field

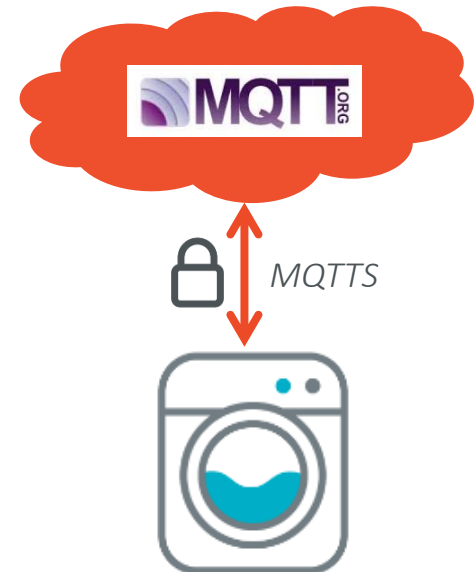


LWM2M (EXTENDED) PROTOCOL STACK



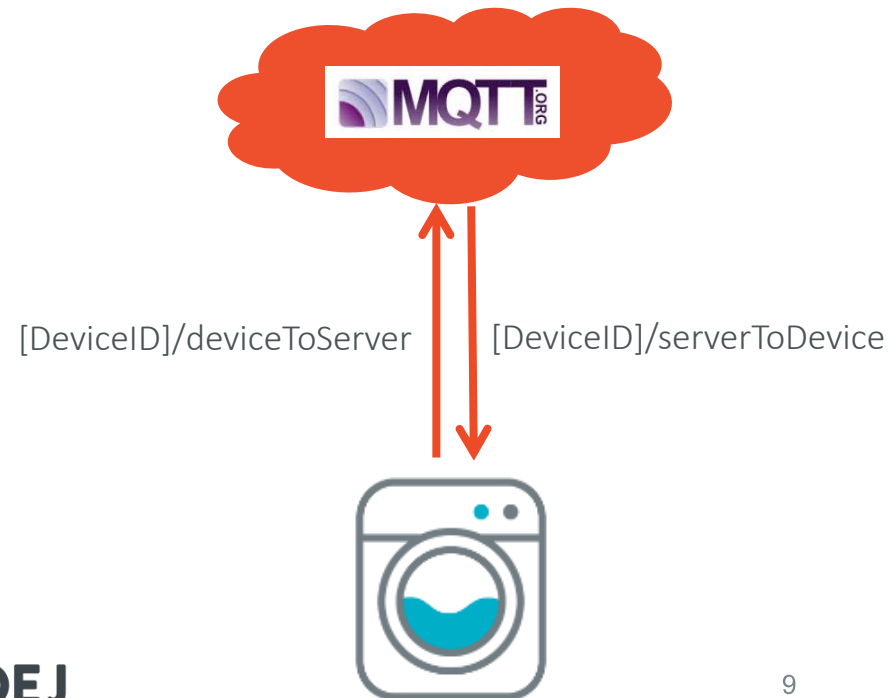
KEY POINT - SINGLE LINK TO THE CLOUD

- For all kind of data transfer
 - Data & Device management
- Security
 - Not Yet Another Link to Secure
 - No additional protocol for large data transfer (firmware upgrade)
- Device Code Footprint
 - Embed only one vertical stack



KEY POINT - DEDICATED TOPICS FOR LWM2M

- 2 general purpose topics: “transport topics”
 - “deviceToServer” for ascending messages
 - “serverToDevice” for descending messages
- Prefixed per device
- Payload = CoAP messages



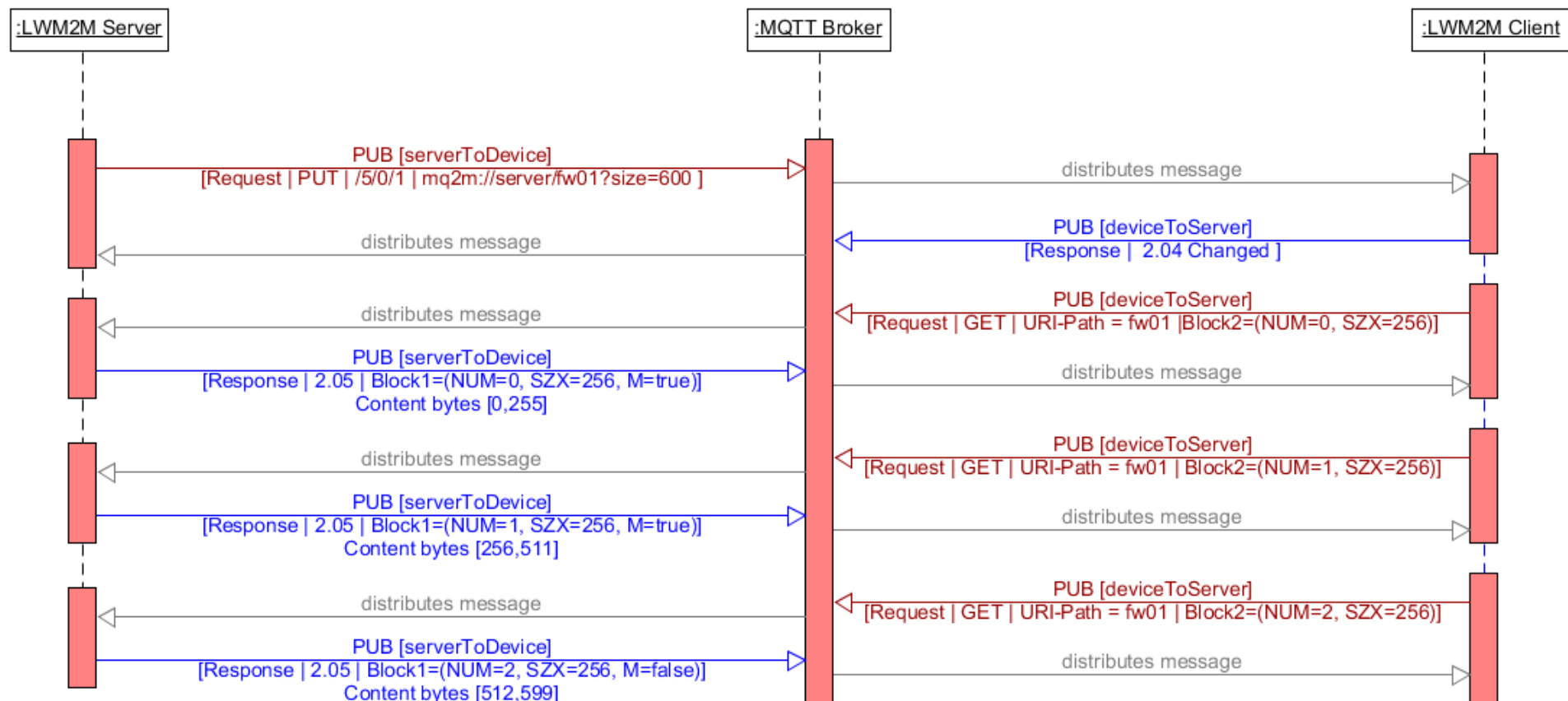
KEY POINT - REMOVE CoAP RELIABILITY STUFF

- Connection is maintained at TCP layer
- No Confirmable (CON) and Acknowledgment (ACK) messages
- No duplication
 - QoS set to 0
 - Retain mode set to false
 - CleanSession flag set to 1
- Only CoAP compact message format is kept

KEY POINT - LARGE BINARY TRANSFER

- Firmware Upgrade / Application Upgrade
- Use of LwM2M Package URI resources (no Package)
- A new URI scheme for fetching a resource
 - `mq2m://[owner]/[path]?size=xxx`
 - `mq2m://server/firmware-1.1.0?size=12345678`
- Device controls the download rhythm/rate/schedule
- Block oriented
 - Using Block-wise Transfer Block Options semantic

LARGE BINARY TRANSFER - EXAMPLE



CURRENT SPECIFICATION HOSTING

- ESR - Publicly available for comments and implementation
- <http://e-s-r.net/download/specification/ESR030-LWM2M-MQTT-1.0-A.pdf>



SPECIFICATION NEXT STEPS

- Ownership
 - Move to appropriate organization OMA
 - Anyone interested can help us on the process ?
- Content
 - Final update against Feb 2017 LwM2M 1.0 release
 - Self-contained specification
 - No CoAP IETF dependency
 - CoAP message format may be derived
 - Proposal for LwM2M specification split
 - LwM2M semantics & core normative objects
 - LwM2M bindings (UDP / SMS / TCP / MQTT...)



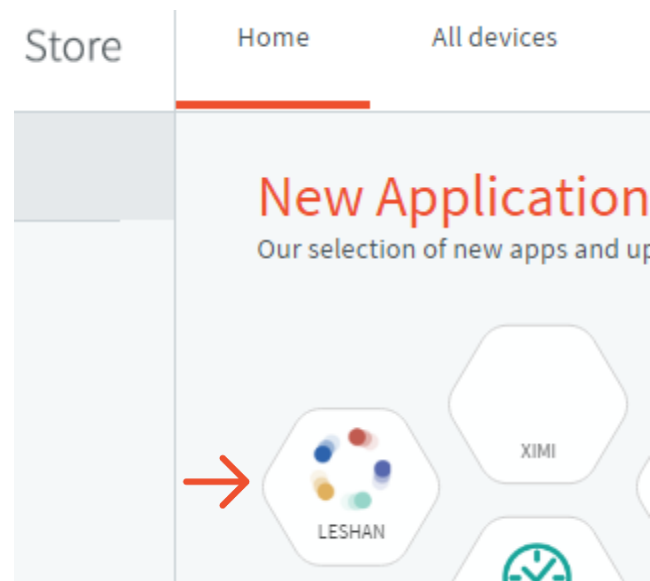
MICROEJ REFERENCE IMPLEMENTATION (1)

- Leshan client port on MicroEJ (ARM Cortex-M)



MICROEJ REFERENCE IMPLEMENTATION (2)

- MicroEJ Leshan Server runs LwM2M over MQTT
 - <http://leshan.microej.com:8080/#/clients>
- MicroEJ Leshan Client App available on MicroEJ Store
 - <https://communitystore.microej.com>



REFERENCE IMPLEMENTATION NEXT STEPS

- Introduced to Eclipse IOT WG call
- Can be delivered as a Leshan sub-project
- Suggest core modifications to main stream to allow to plug this new backend

FREQUENTLY ASKED QUESTIONS

- Why not LwM2M over CoAP over TCP ?
 - Reduced CoAP scope to only cover the LwM2M case
 - Would break the assumption to keep a single secured link for everything
- Why over MQTT ?
 - Assumption that MQTT is a requirement
- What about IETF “draft-koster-core-coap-pubsub-05” ?
 - This is a proposal for a pub/sub paradigm on top of CoAP



THANK YOU

FOR YOUR ATTENTION!

