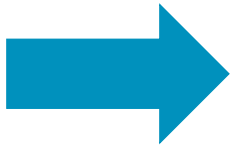
Joint IETF/IRTF – OMA Meeting

LwM2M Tutorial

Hannes Tschofenig

19th July 2019
Montreal/Canada

Agenda



Architecture and Protocol Stack

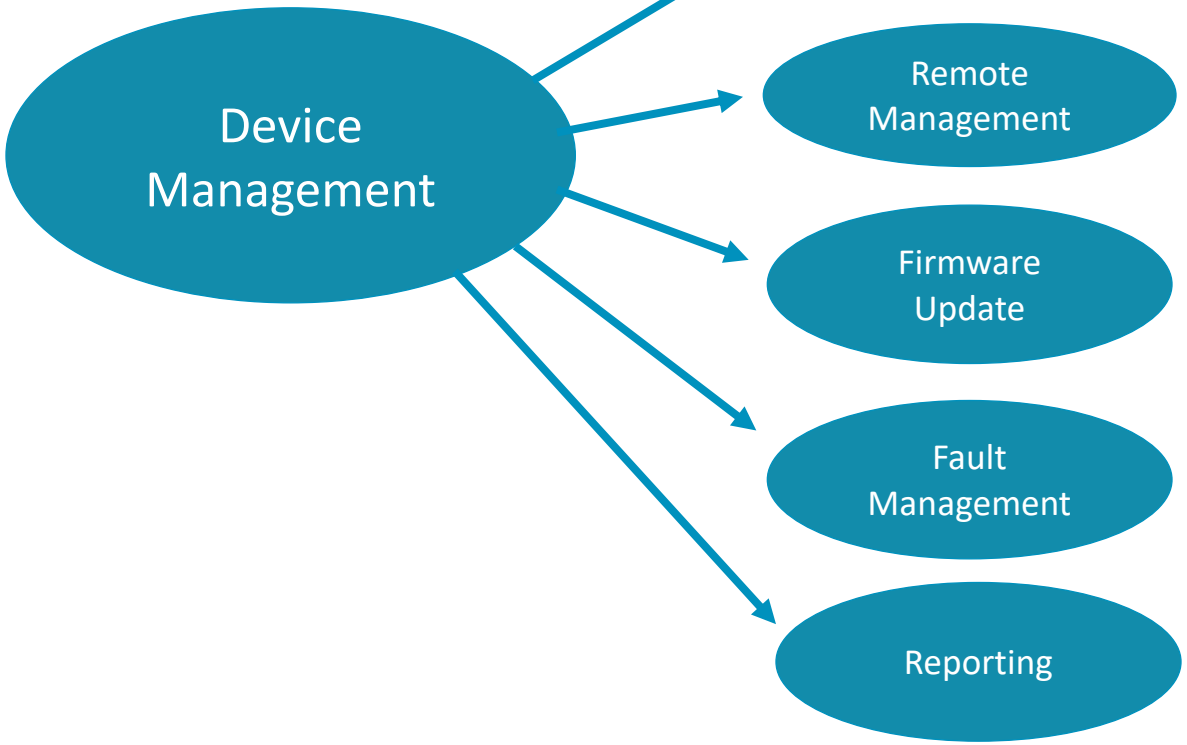
Messaging Layer

Data model

Securing LwM2M

Bootstrapping

What is IoT device management?



- Key management
- Service provisioning
- Provisioning of access control lists

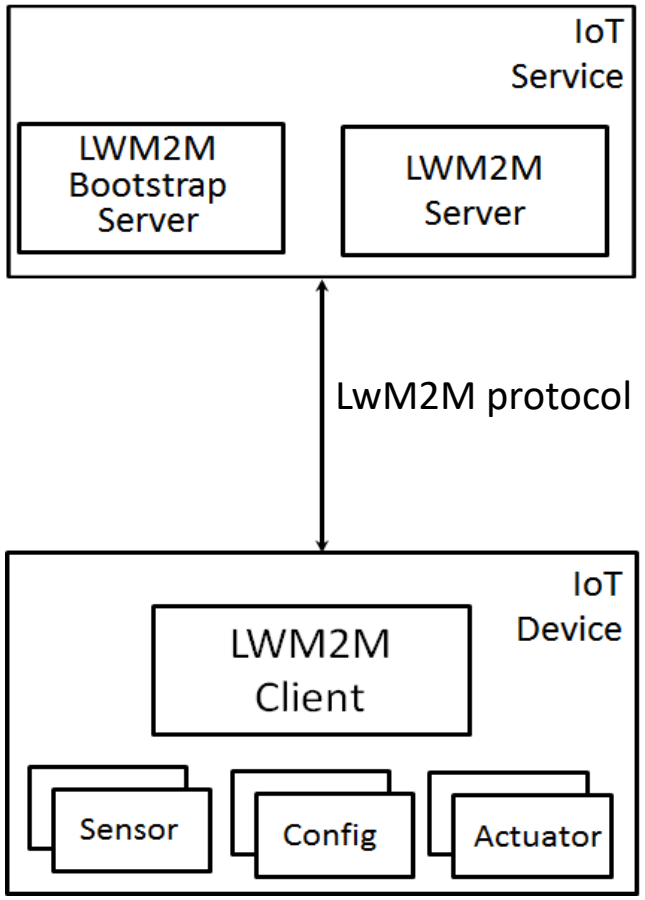
- Changes to settings
- Trigger actuators

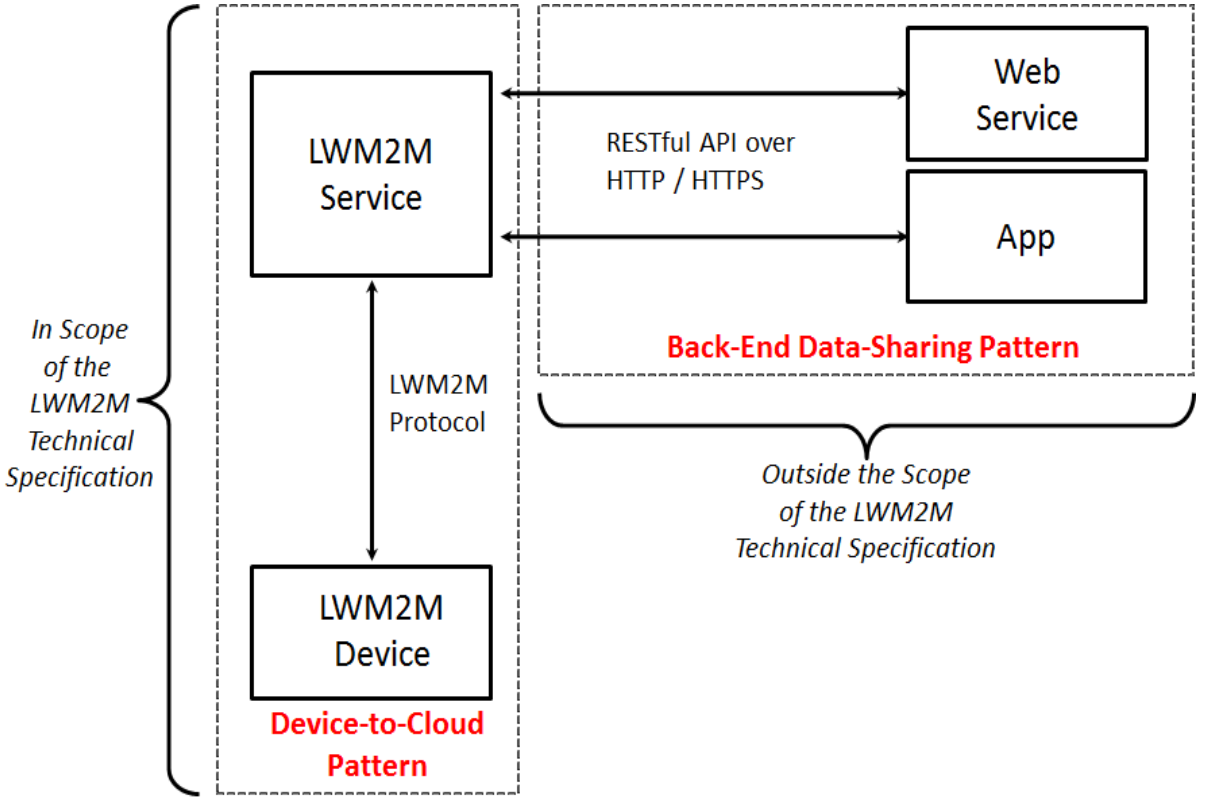
- Update application and system software
- Apply bug fixes and add new features

- Report errors from devices
- Query status of devices

- Notify changes in sensor values
- Retrieve configuration settings and device status

LwM2M Architecture





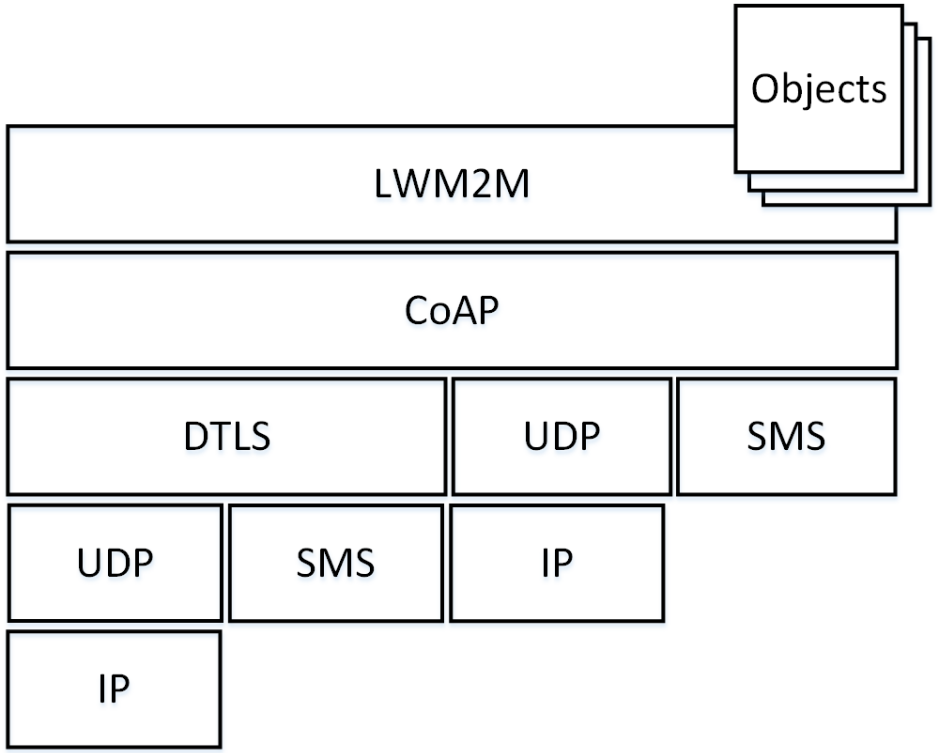
For an introduction on the IoT design patterns see at http://www.internetsociety.org/doc/iot-overview

LwM2M v1.1.1

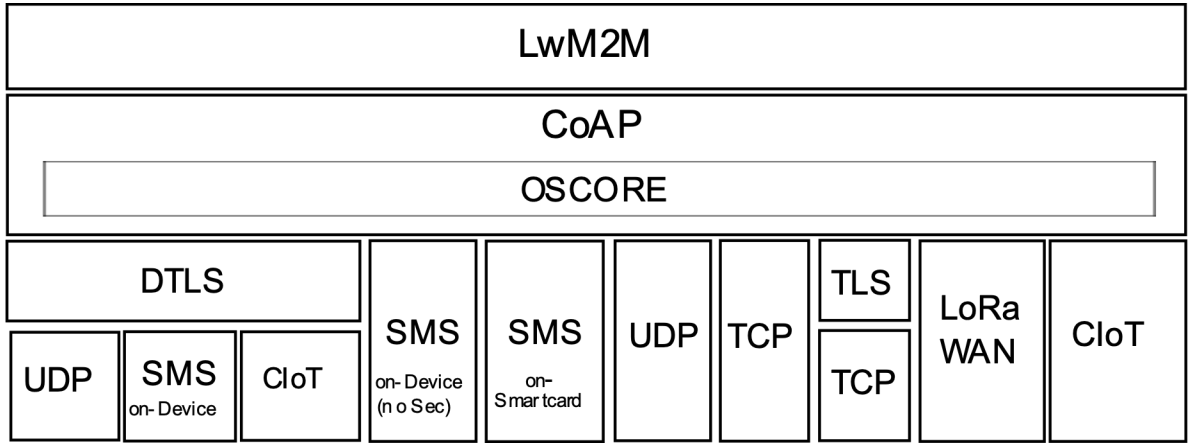
- We recently published version 1.1.1
  - v1.1.1 is a bugfix release for v1.1. v1.1 was published June 2018.
- Specifications are available for download here: https://www.openmobilealliance.org/release/LightweightM2M/V1_1_1-20190617-A/
  - Transport spec
  - Core spec

Protocol Stack

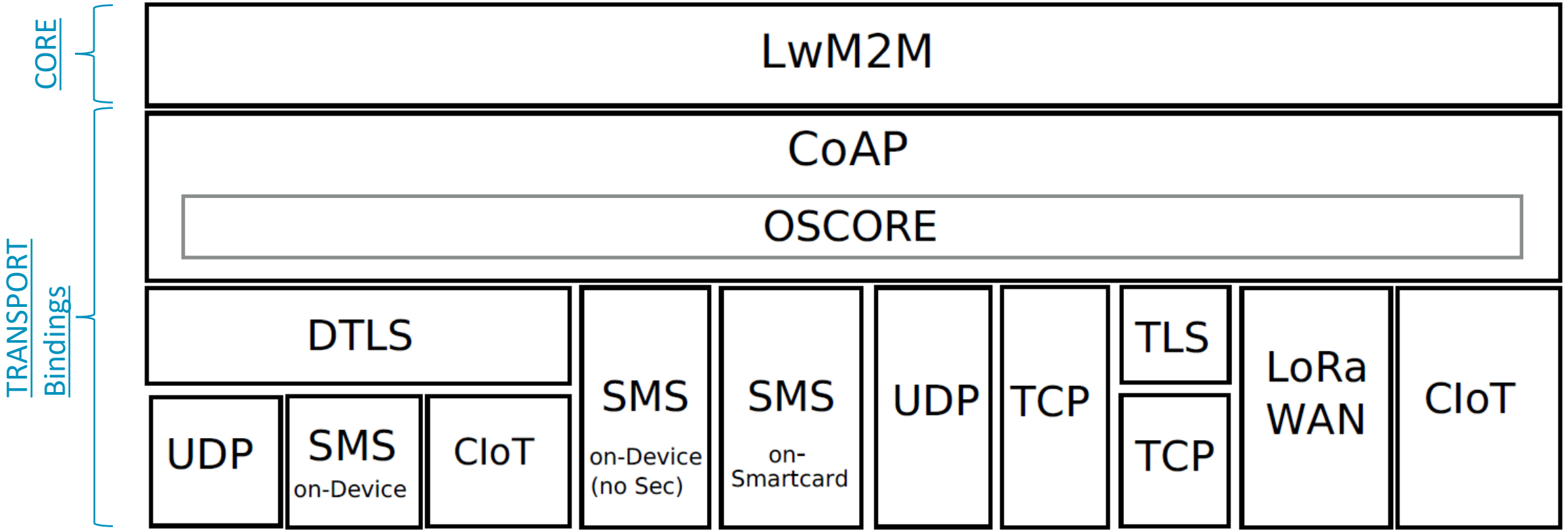
LwM2M v1.0, v1.0.1, v1.0.2



LwM2M v1.1.1



LwM2M v1.1.1


CORE
TRANSPORT
Bindings
LwM2M
CoAP
OSCORE
DTLS
UDP
SMS
on-Device
CIoT
SMS
on-Device
(no Sec)
SMS
on-
Smartcard
UDP
TCP
TLS
TCP
LoRa
WAN
CIoT

Low Power WAN Support

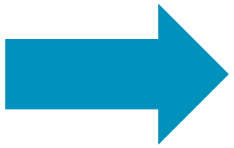
- Currently supported transports are LoRaWAN and 3GPP CIoT (NB-IoT and LTE-M).
- Specification offers generic introduction to technologies and their topologies, new terminology, recommendations for timer settings, and defines transport bindings.
- LoRaWAN is specified to
  - operate in NoSec mode (i.e., with LoRaWAN link layer security)
  - The LwM2M Server and the LwM2M Bootstrap-Server are LoRaWAN Application Servers.
  - The "register" operation has different parameter requirements. None of the parameters are mandatory. The Endpoint Client Name is not used.
  - CoAP transmissions were adapted to LoRaWAN usage specifics
- 3GPP CIoT (NB-IoT and LTE-M) follows “plain” LwM2M and allows use of DTLS. No changes to CoAP are necessary.

New in LwM2M v1.1: CoAP over TCP/TLS

> Quote:
>
> "The primary reason for introducing CoAP over TCP [RFC793] and TLS [RFC5246] is that some networks do not forward UDP packets. Complete blocking of UDP happens in between about 2% and 4% of terrestrial access networks, according to [EK2016]. UDP impairment is especially concentrated in enterprise networks and networks in geographic regions with otherwise challenged connectivity." RFC 8323

- Positive side-effects of using TCP:
  - Fewer keep-alive messages. According to [HomeGateway], the mean for TCP and UDP NAT binding timeouts is 386 minutes (TCP) and 160 seconds (UDP).
  - TCP utilizes mechanisms for congestion control and flow control that are more sophisticated than the default mechanisms provided by CoAP over UDP. CoAP block-wise transport still useful.
- TCP has more overhead than UDP but there is room to tweak TCP.

Agenda



Architecture and Protocol Stack

Messaging Layer

Data model

Securing LwM2M

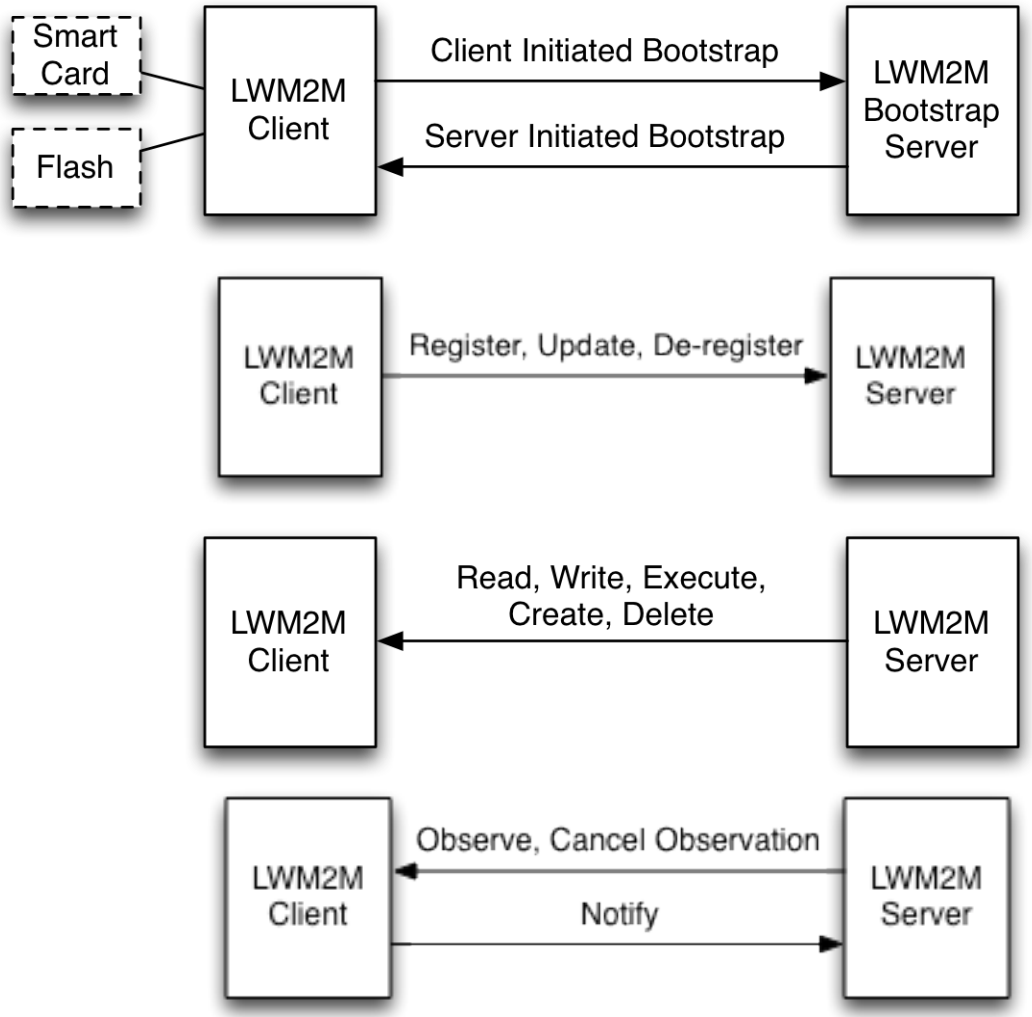
Bootstrapping

LwM2M v1.2

LwM2M Messaging Layer

High-level message pattern (using RESTful API) hiding details of underlying transports

- Bootstrap interface
  - Configure credentials and ACLs for LwM2M servers
- Registration interface
  - Informs server about “existence” of the LwM2M device and supported functionality (e.g., objects, transport bindings)
- Device management & service enablement interface
  - Ability to access object instances and resources
- Information reporting interface
  - Publish/subscribe interaction for observing changes in resources.



LwM2M v1.0

Example: Registration

CoAP POST

Confirmable, Message ID: 18947

Token: 0a0f255fe71b37a0

Opt Name: #1: Uri-Path: rd

Opt Name: #2: Content-Format: application/link-format

Opt Name: #3: Uri-Query: b=US ← Binding Modes

Opt Name: #4: Uri-Query: lwm2m=1.1 ← LwM2M Version

Opt Name: #5: Uri-Query: lt=600 ← Lifetime

Opt Name: #6: Uri-Query: ep=john-VirtualBox ← Endpoint Client Name

Payload: Payload Content-Format: application/link-format

</>;rt="oma.lwm2m", </1/0>, </3/0>, </6/0>, </3303/0>

← Objects and Object Instances

CoAP Ack

2.01 Created, Message ID:18947

Token: 0a0f255fe71b37a0

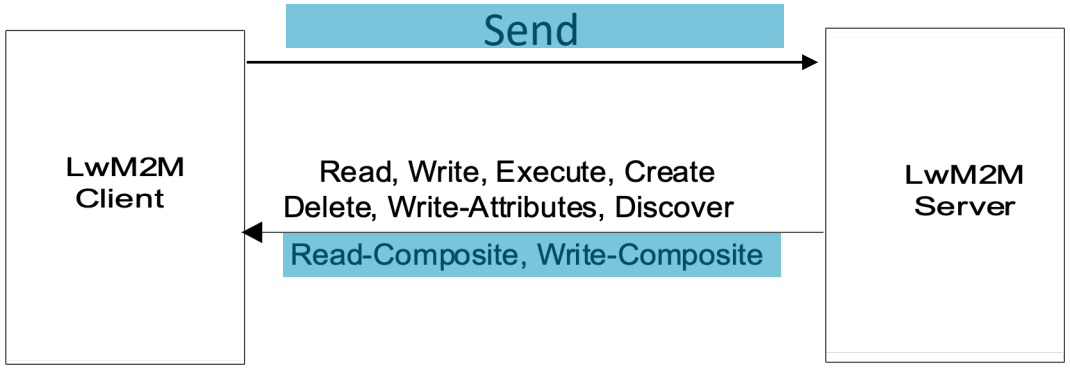
Opt Name: #1: Location-Path: rd

Opt Name: #2: Location-Path: Chohyv1bmr

Device Management & Service Enablement Interface

v1.1.1

API Description



LwM2M v1.1 Enhancements

"Read-Composite" operation used to selectively read a number of Resources, and/or Resource Instances of different Objects in a single request.

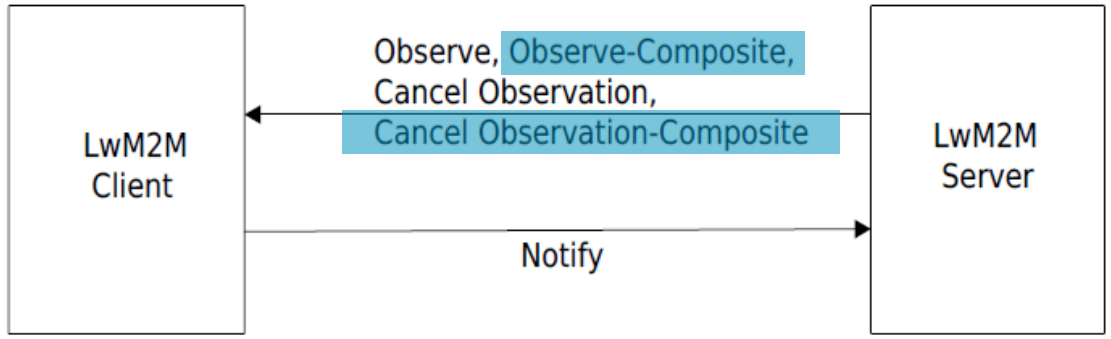
- More efficient than repeatedly sending requests to different objects/resources.

The "Send" operation is used by the LwM2M Client to send data to the LwM2M Server unsolicited.

Information Reporting Interface

v1.1.1

API Description

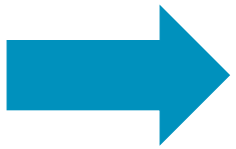


LwM2M v1.1 Enhancements

Information reporting interface supports a publish-subscribe model. New optimizations were introduced with v1.1:

- The "Observe-Composite" operation to initiate observations for a group of resources and/or resource instances across multiple object instances.
- Each resource can have multiple observe conditions attached (configured using the “Write-Attribute” command).

Agenda



Architecture and Protocol Stack

Messaging Layer

Data model

Securing LwM2M

Bootstrapping

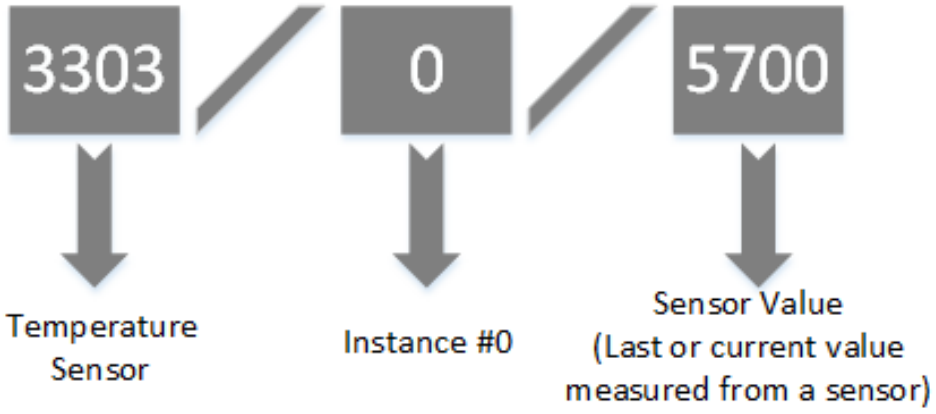
LwM2M v1.2

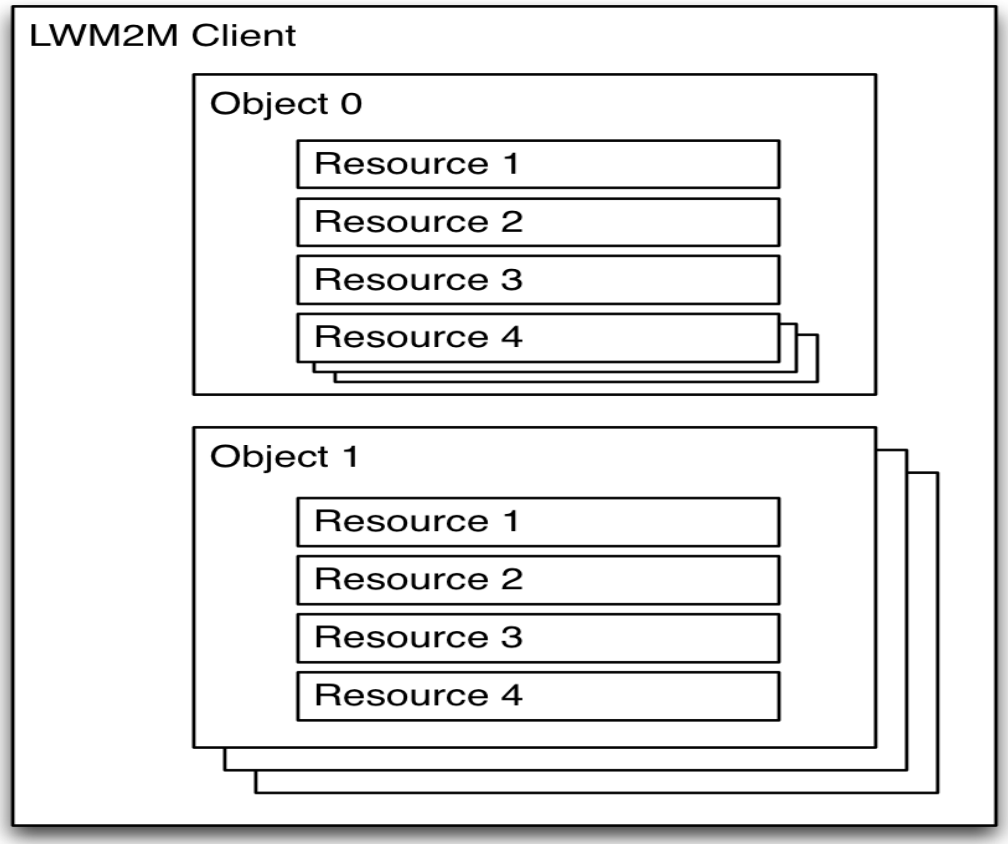
Data Model in LwM2M

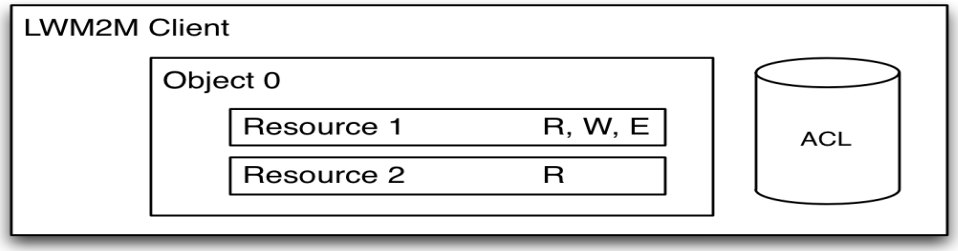
Objects/Resources are accessed with simple URIs:

/{Object ID}/{Object Instance}/{Resource ID}

Example:







Objects Versions !?#$

Concept

Idea was to avoid registering a new object id every time an object is modified

Allow object evolution independent from “enabler” specification (for those that are found in the core specification)

Object versions are signaled in the registration message

- Implicit signal for objects defined in a given enabler

Version 1.1

URN / Version	XML Name	LwM2M Editor	Object Name	Technical Specification
urn:oma:lwm2m:oma:0	0	0	LWM2M Security	
urn:oma:lwm2m:oma:0:1.1	0	0	LWM2M Security	
urn:oma:lwm2m:oma:1	1	1	LwM2M Server	
urn:oma:lwm2m:oma:1:1.1	1	1	LwM2M Server	
urn:oma:lwm2m:oma:2	2	2	LwM2M Access Control	
urn:oma:lwm2m:oma:3	3	3	Device	
urn:oma:lwm2m:oma:4	4	4	Connectivity Monitoring	
urn:oma:lwm2m:oma:4:1.1	4	4	Connectivity Monitoring	

Data access in LwM2M v1.1

Extension to enable resource instance level access:

/{Object ID}/{Object Instance}/{Resource ID}/{Resource Instance}

Device Object Example

Resource Name	Resource ID	Resource Instance ID	Value	Notes
Manufacturer	0		Open Mobile Alliance	
Model Number	1		Lightweight M2M Client	
Serial Number	2		345000123	
Firmware version	3		1.0	
Available Power Sources	6	0	1	Internal Battery
Available Power Sources	6	1	5	USB

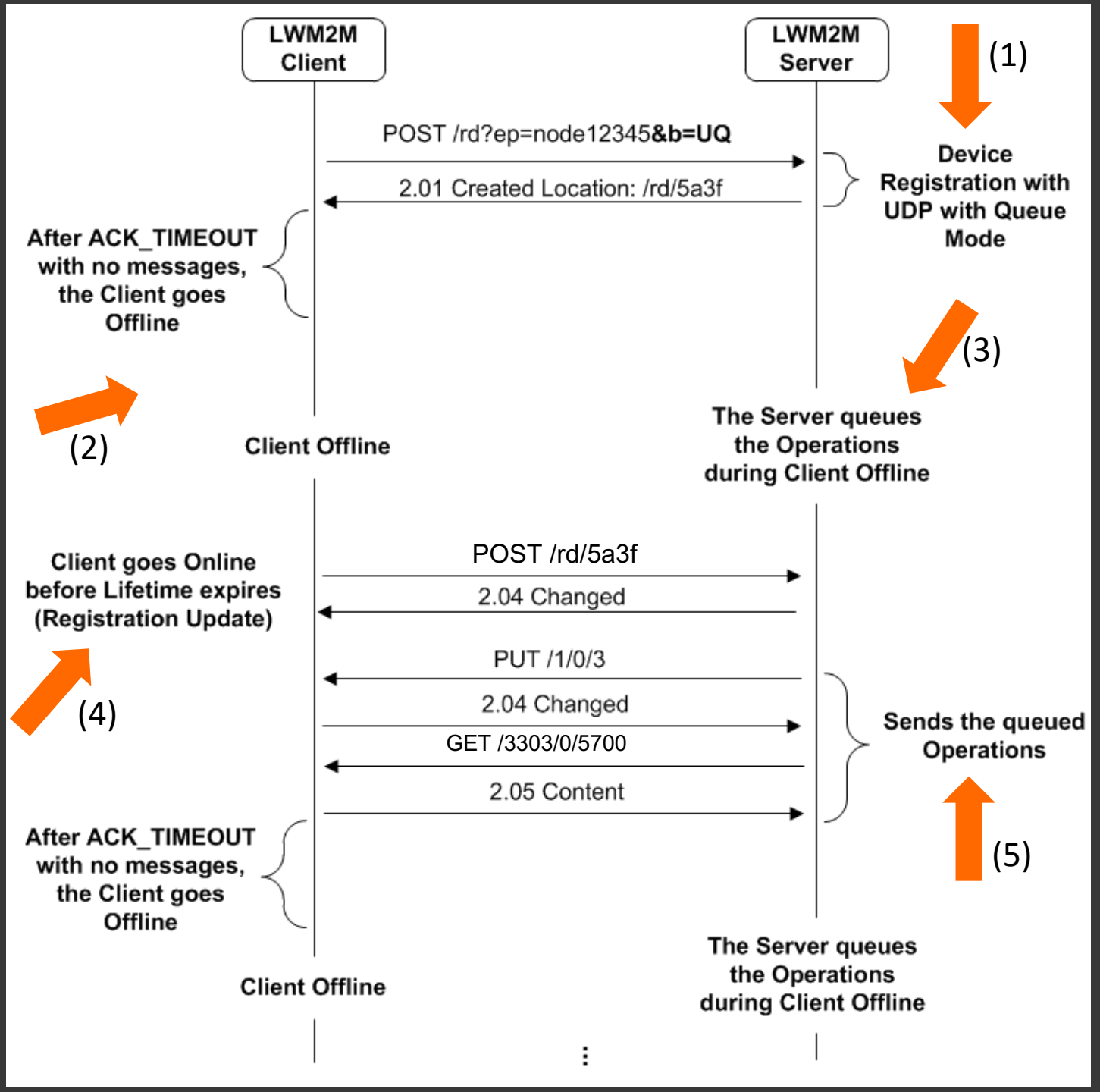
Objects

The LwM2M specification itself defines only eight objects.

Object Name	ID	Description
LwM2M Security	0	Keying material of a LwM2M Client to access a LwM2M Server/Bootstrap-Server.
LwM2M Server	1	Data related to a LwM2M Server.
Access Control	2	Information used to check whether a LwM2M Server has access to a specific object.
Device	3	Device related information, including device reboot and factory reset functions.
Connectivity Monitoring	4	Parameters related to network connectivity.
Firmware Update	5	Capability to update firmware
Location	6	Device location information
Connectivity Statistics	7	Information like transmit and receive counters

Energy Efficiency - Queue Mode

- Easy to interact with devices that are always connected.
- For energy efficiency reasons many IoT devices sleep most of the time.
- Client uses the registration refresh message to inform LwM2M server that it is awake and ready to receive messages.
- Server conveys messages to client within a given time window.



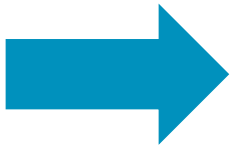
24

Agenda

Architecture and Protocol Stack

Messaging Layer

Data model



Securing LwM2M

Bootstrapping

Securing LwM2M v1.0

IoT devices need communication security.

V1.0 relied on DTLS and supported **three credential types**:

- Pre-shared secrets,
- Raw public keys, and
- Certificates.

Recommended algorithms:

- AES-128-CBC, and AES-128-CCM / AES-128-CCM-8
  (for symmetric crypto)
- Elliptic Curve Cryptography (ECC)
  (for asymmetric crypto)

Constrained IoT Device

Less-constrained IoT Device

See webinars on TLS / DTLS concepts.

Securing LwM2M v1.1/v1.1.1

- Uses “DTLS / TLS Profiles for IoT” (RFC 7925) with lots of recommendations.
- New resource to separate the SNI from the LwM2M Server URI. This makes deployments where the certificate content and the URIs do not match practical.
- Guidelines for use of certificate revocation and certificate expiry
- Error handling procedures for dealing with TLS/DTLS errors.
- Still offers AES-CBC but recommended not to be used.
- Enhanced PKI functionality (next slide)
- Added OSCORE support

Enhanced PKI Support

- In **LwM2M v1.0** the certificate in the Security Object associated with the LwM2M / Bootstrap Server had to be the certificate used by the server. This allowed only **certificate pinning**.
- LwM2M v1.1 added support for additional PKI deployment models, including
  - “CA constraint” because it limits which CA can be used to issue certificates.
  - “Service Certificate Constraint" limits which end entity certificate can be used. Must pass PKIX validation.
  - “Trust Anchor Assertion” is used to specify a certificate that MUST be used as the trust anchor when validating the end entity certificate given by the server in TLS.
  - “Domain-Issued Certificate" limits which end entity certificate can be used. PKIX validation is not used.
- Supports also raw public keys and fingerprints of certificates.

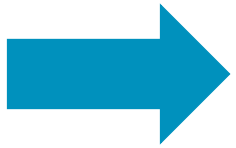
Agenda

Architecture and Protocol Stack

Messaging Layer

Data model

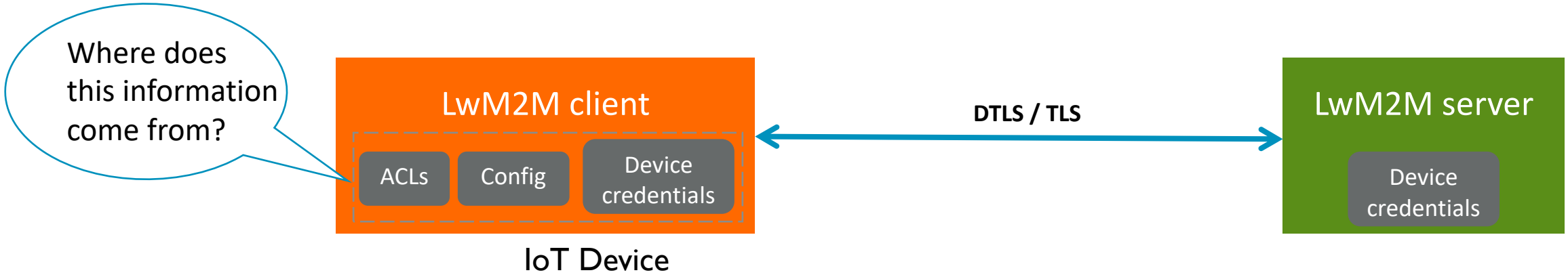
Securing LwM2M



Bootstrapping

Bootstrapping

A LwM2M client needs credentials to securely communicate with the LwM2M server. Configuration and access rights might also change over time.



Information comes from a trusted third party and specification offers several deployment choices (including factory bootstrap, client initiated bootstrap and server initiated bootstrap).

IPSO Alliance whitepaper compares the different bootstrapping (credential management) approaches.

Bootstrapping

(Supported since LwM2M v1.0)

Server-generated Credentials

Used for symmetric keys and raw public keys.

Can also be used for certificates (with the argument that IoT devices offer a weak random number generator)

Bootstrap server generates the keying material and sends it to the client.

Communication is protected using DTLS/TLS.

Client-generated Credentials

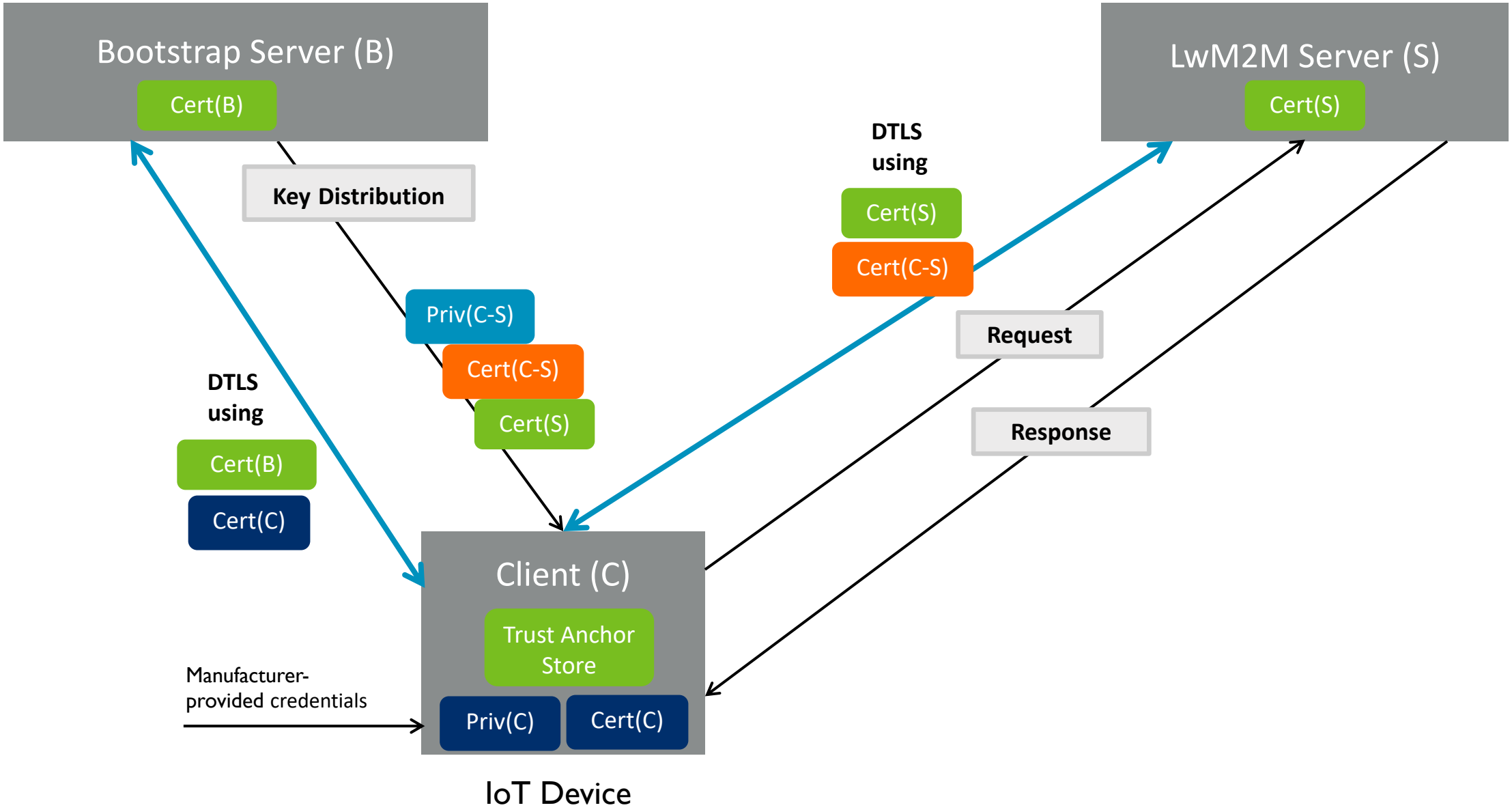
Available with “Enrollment over Secure Transport (EST)” over CoAP

Private key remains on the IoT device and Certificate Signing Request is protected using DTLS/TLS.

- Added value from a security point of view

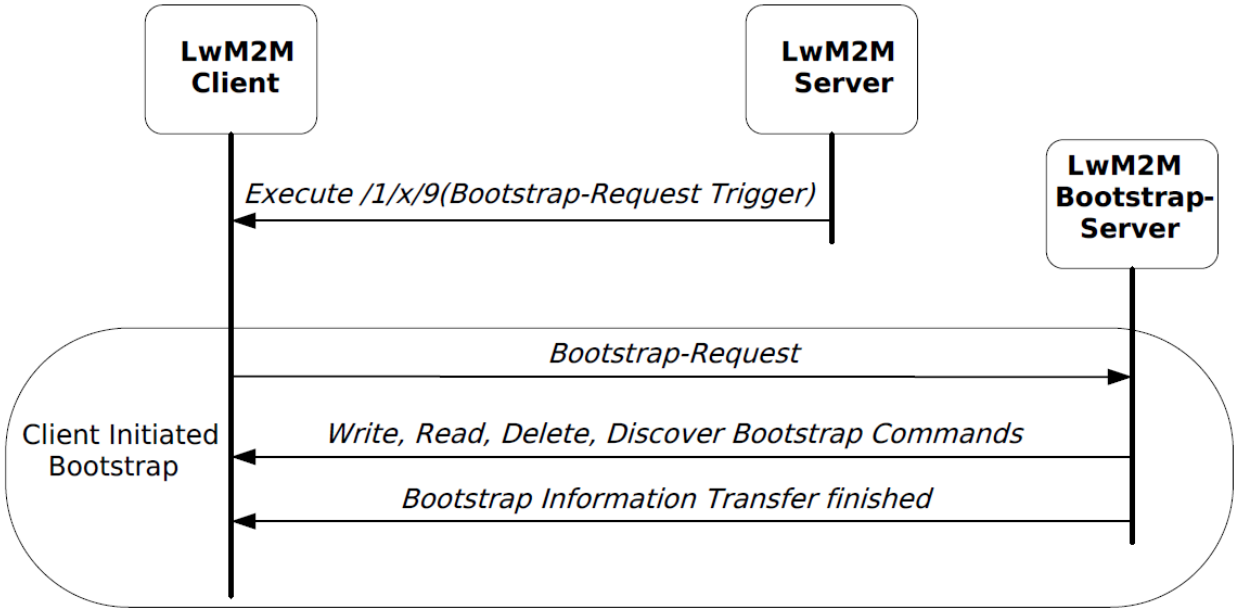
31

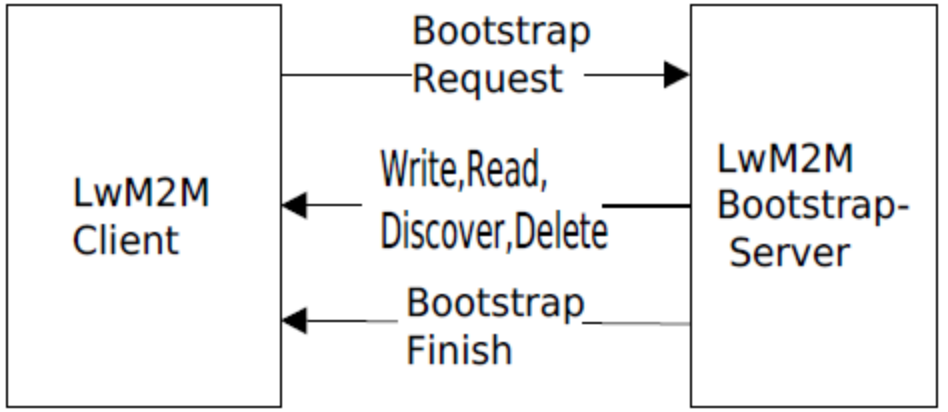
Example: Server-Generated Certificate Bootstrapping



Server-Initiated Bootstrapping (LwM2M v1.1)

- Server-initiated bootstrapping caused lots of problems in LwM2M v1.0
  - It required a role reversal and the IoT device to act as a DTLS/TLS server.
- Concept only worked when the server has reachability information about the IoT device.
  - Does not work when the device is behind an unmanaged firewall or behind a NAT.
- Replaced with the ability to allow a dedicated LwM2M server to trigger a client-initiated bootstrapping.
- Added support for incremental bootstrapping.





Agenda

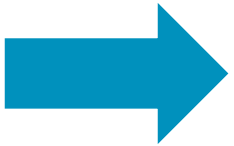
Architecture and Protocol Stack

Messaging Layer

Data model

Securing LwM2M

Bootstrapping



Summary

- LwM2M re-uses IETF stack, including security mechanisms.
- We actively contribute to the IETF standardization (e.g., SenML Fetch/Patch, CoAP over TCP/TLS).
- LwM2M is implemented by OMA members and also non-members.
- 7 testfests already took place and next one will be in Oct. 2019 in Seoul. Spec is in development for 5+ years.
- LwM2M has successfully been used over IP- and non-IP-based transports (LoRaWAN & NB-IoT).
- LwM2M was designed with low-end IoT devices and constrained networks in mind. LwM2M v1.1.1 added even more performance improvements.