ERICSSON

OMA LIGHTWEIGHT M2M V1.0

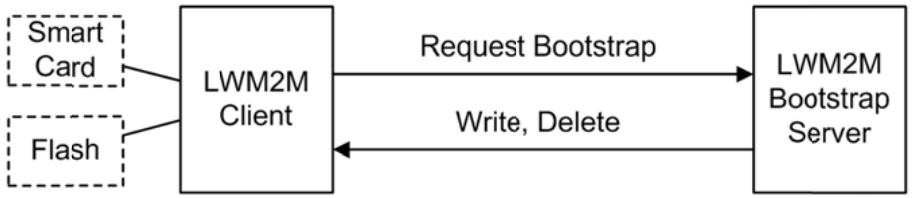
LWM2M

Jaime Jiménez

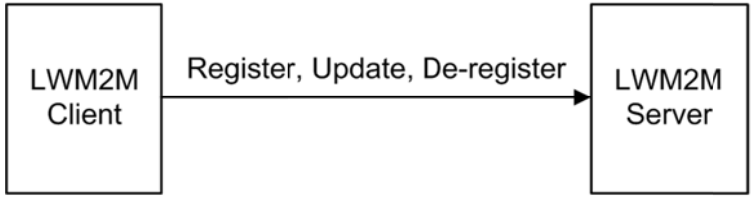
TABLE OF CONTENTS

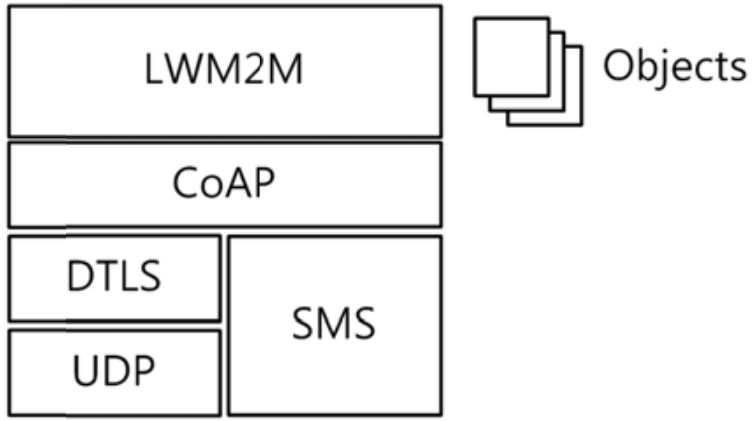
INTRODUCTION

- There are 4 interfaces:
  1. Bootstrap



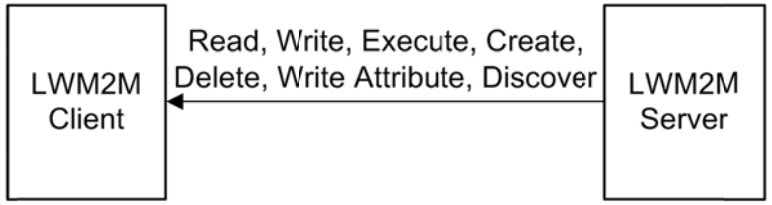
  2. Client Registration



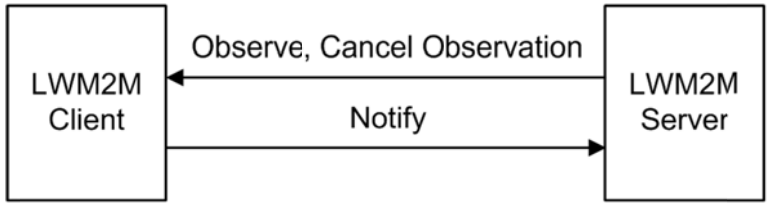


INTRODUCTION

- There are 4 interfaces:
  3. Device Management



  4. Information Reporting



Interface	Direction	Operation
Bootstrap	Uplink	Request Bootstrap
Bootstrap	Downlink	Write, Delete
Client Registration	Uplink	Register, Update, De-register
Device Management and Service Enablement	Downlink	Create, Read, Write, Delete, Execute, Write Attributes, Discover
Information Reporting	Downlink	Observe, Cancel Observation
Information Reporting	Uplink	Notify

BOOTSTRAPPING

- There are 4 bootstrap modes:
  - Factory Bootstrap (everything preconfigured)
  - Bootstrap from Smartcard (information read from a smartcard, requires a secure channel btw smartcard and device)
  - **Client initiated Bootstrap** (used when server is not configured in the client or initial config fails, the LWM2M Bootstrap Server will give the information. The server will use the write/ delete to modify the information)
  - Server Initiated Bootstrap (Just like before, but without the client sending request. The server already knows that the client is ready. How the server knows that is implementation specific.)

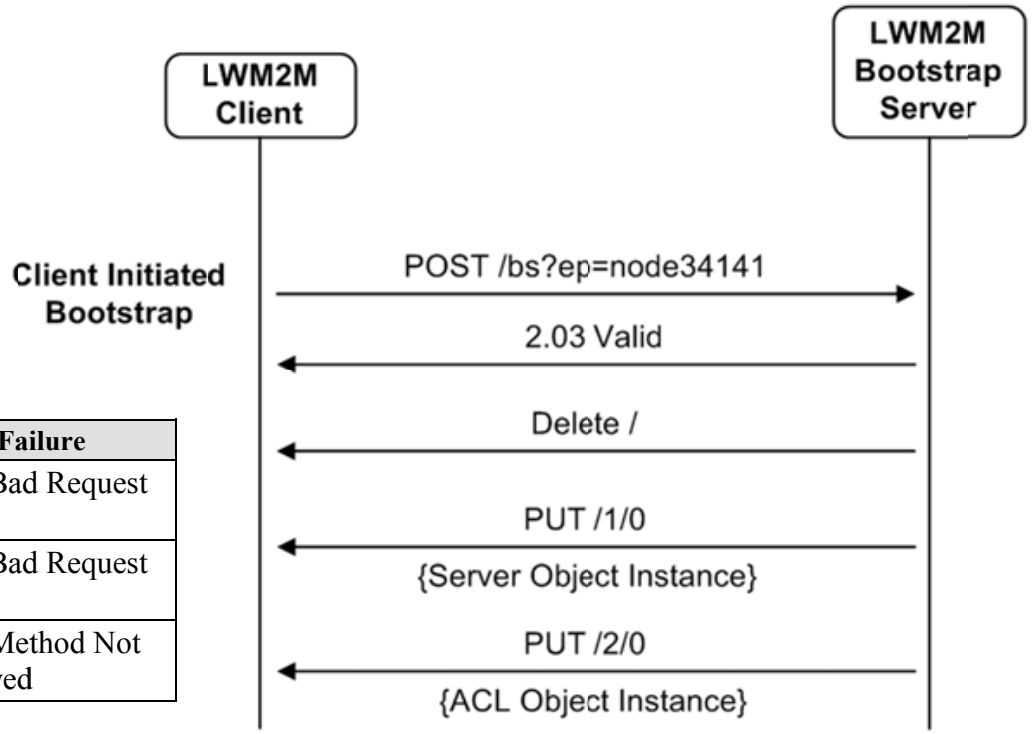
BOOTSTRAPPING

- The bootstrap sequence is the following:
  1. If device has Smartcard, the device tries to use it.
  2. If not, it tries the factory bootstrap.
  3. If none of the above and the server has LWM2M Server Object instances from the previous instances, then the device tries to register using them.
  4. If all those steps fails, and the client does not receive the server initiated bootstrap within the ClientHoldOffTime then it uses the client initiated bootstrap. The client must have the LWM2M Server Bootstrap Information (with at least one LWM2M Server account) after this step.

BOOTSTRAPPING: EXAMPLE

› Client Initiated Bootstrap

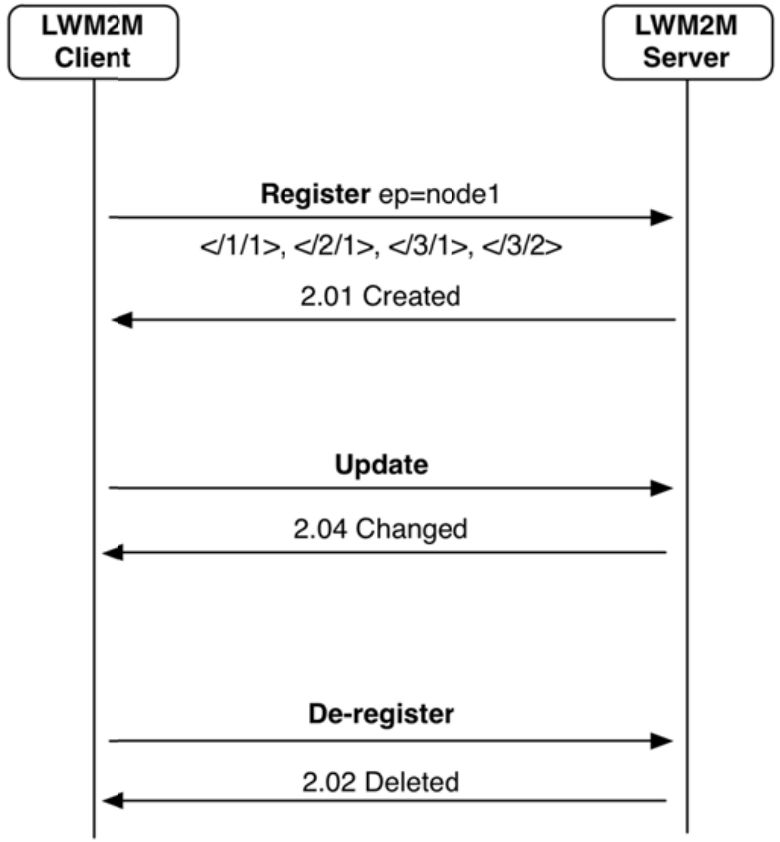
1. Client requests bootstrap, sending its endpoint name
2. Server accepts and deletes existing content on device(“/” URI)
3. Server creates Object Instance for LWM2M Server (/1/0)
4. Server creates an Access Control Instance (/2/0)



Operation	CoAP Method	URI	Success	Failure
Request Bootstrap	POST	/bs?ep={Endpoint Client Name}	2.04 Changed	4.00 Bad Request
Write	PUT	/{Object ID}/{Object Instance ID}/ {Resource ID}	2.04 Changed	4.00 Bad Request
Delete	DELETE	/{Object ID}/{Object Instance ID}	2.02 Deleted	4.05 Method Not Allowed

REGISTRATION

- Register
  - Client provides properties that the server needs to contact the client and maintain the session (e.g. EP name, lifetime, queue mode…) as well as objects in client.
- Update
  - Periodically done, updates lifetime of registration.
- De-register
  - When shutting down device or discontinuing the service.



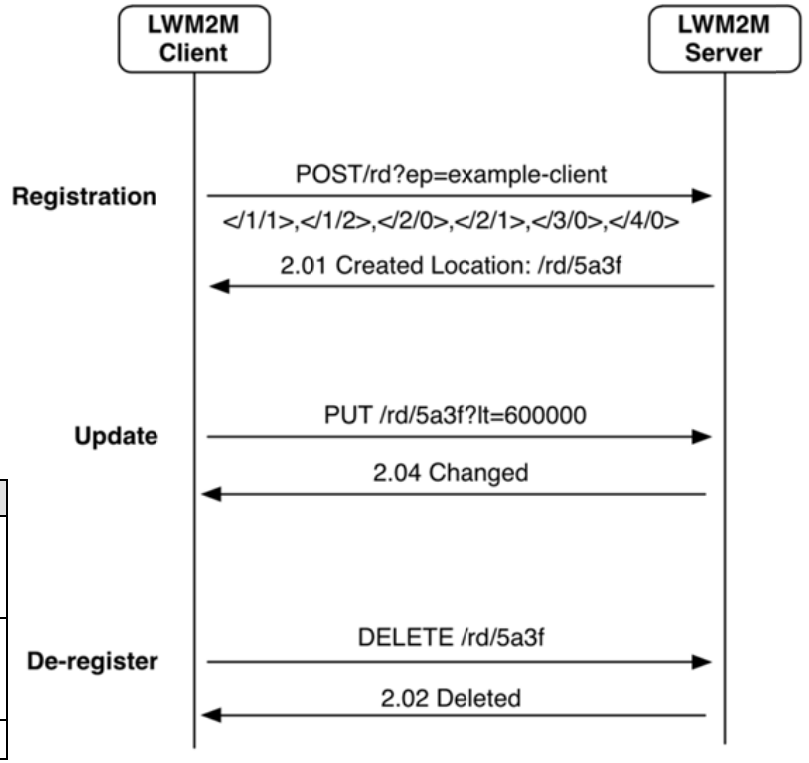
REGISTRATION - REGISTER

- After bootstrapping, LWM2M initiates “Register” operation with the LWM2M Server. Parameters are:
  - Endpoint Client name in URN format (e.g. “urn:imei:0800200c9a66”)
  - *Lifetime (default 86400 seconds)*
  - *LWM2M Version (If > 1.0)*
  - *Binding Mode. Either UDP(U) or SMS (S) And either Serial (S) or Queue (Q) modes. Queue mode means that the server queues all request to client while client is asleep.*
  - *SMS Number (MSISDN) if SMS binding*
  - Objects and instances

REGISTRATION: EXAMPLE

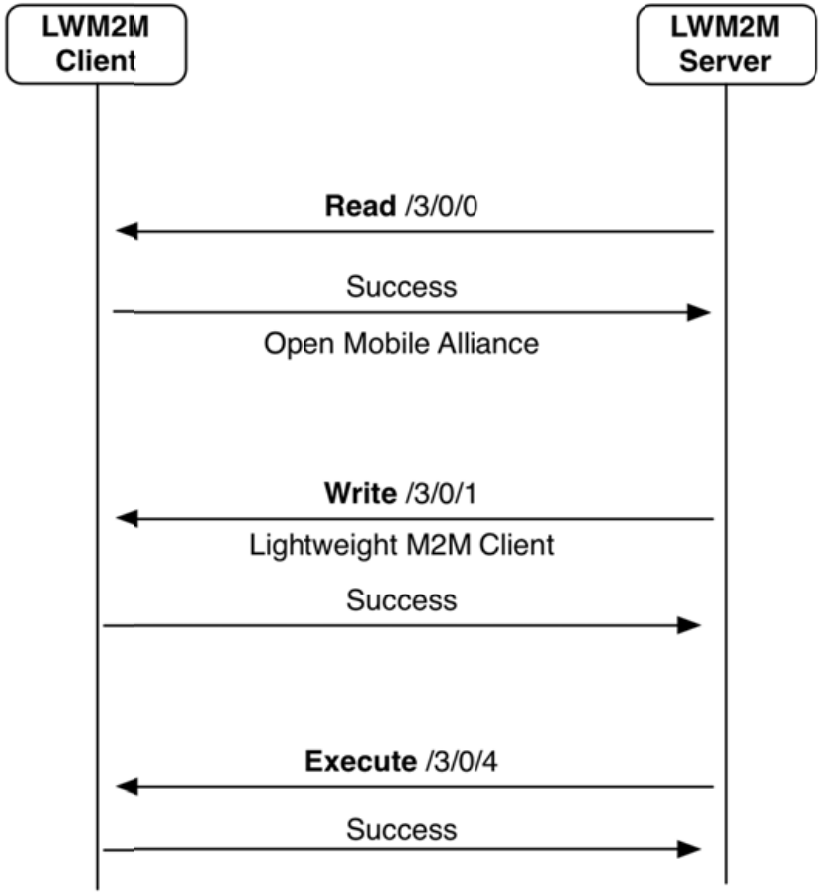
- Registration uses CoAP message types.
  - Client Registers and specifies endpoint, lifetime, version and binding types.
  - Server confirms the creation in Resource Directory.
  - Client Updates modifies the lifetime.
  - Client Deletes entry in Resource directory.

Operation	CoAP Method	URI	Success	Failure
Register	POST	/rd?ep={Endpoint Client Name}&lt={Lifetime}&sms={MSISDN} &lwm2m={version}&b={binding}	2.01 Created	4.00 Bad Request, 4.09 Conflict
Update	PUT	/{location}?lt={Lifetime}&sms={MSISDN} &b={binding}	2.04 Changed	4.00 Bad Request, 4.04 Not Found
De-register	DELETE	/{location}	2.02 Deleted	4.04 Not Found



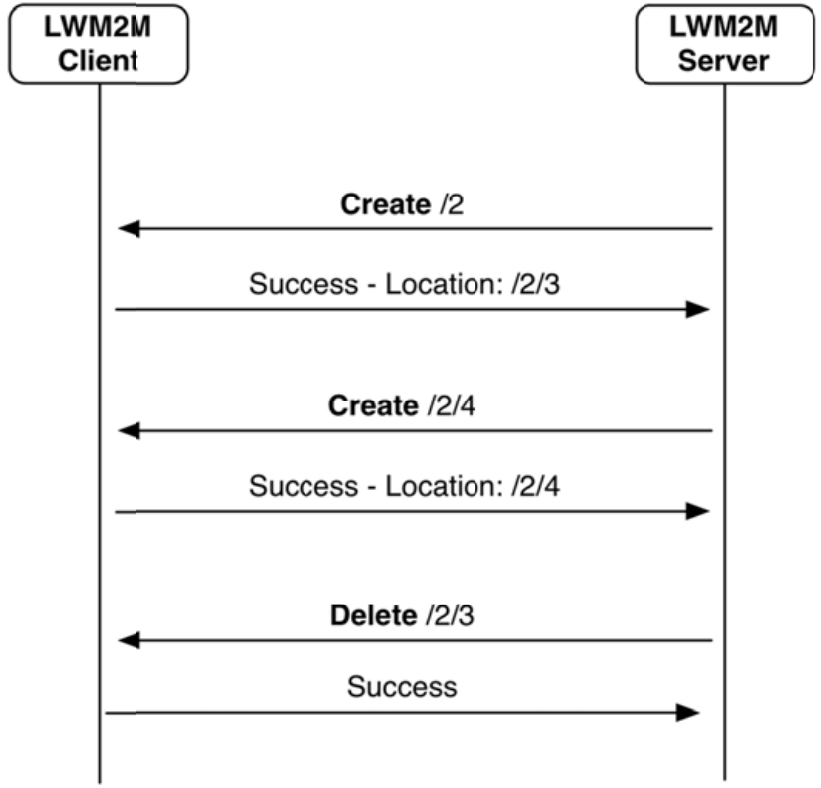
DEVICE MANAGEMENT & SERVICE ENABLEMENT

- Read
  - Client provides properties that the server needs to contact the client and maintain the session (e.g. EP name, lifetime, queue mode…) as well as objects in client.
- Discover (RFC6690)
  - Using CoAP, GET to "/.well-known/core" returns a payload in the CoRE Link Format
- Write
  - Used to change value of Objects & Resources.



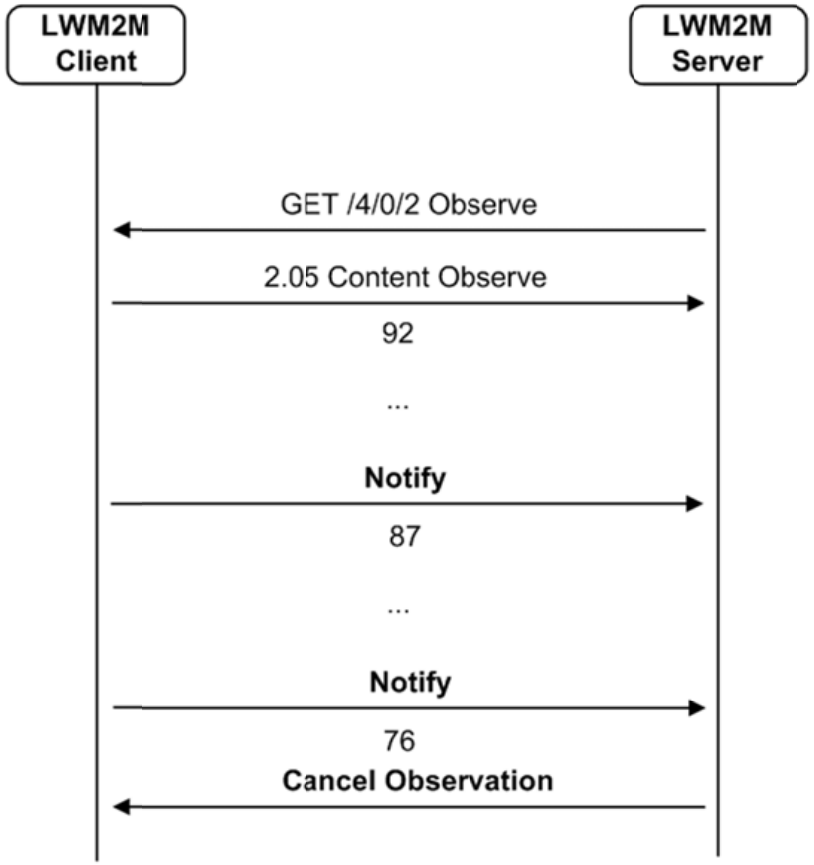
DEVICE MANAGEMENT & SERVICE ENABLEMENT

- Write Attributes
  - These are use with the Observe/Notify operations, wher changes on readable resources occur.
- Execute
  - Initiate action on individual resources.
- Create
  - Used to create and update objects. LWM2M Server will be the owner and has full access rights to the resource.
- Delete
  - Used when the LWM2M Server deletes an Object Instance within the LWM2M Client.



INFORMATION REPORTING INTERFACE

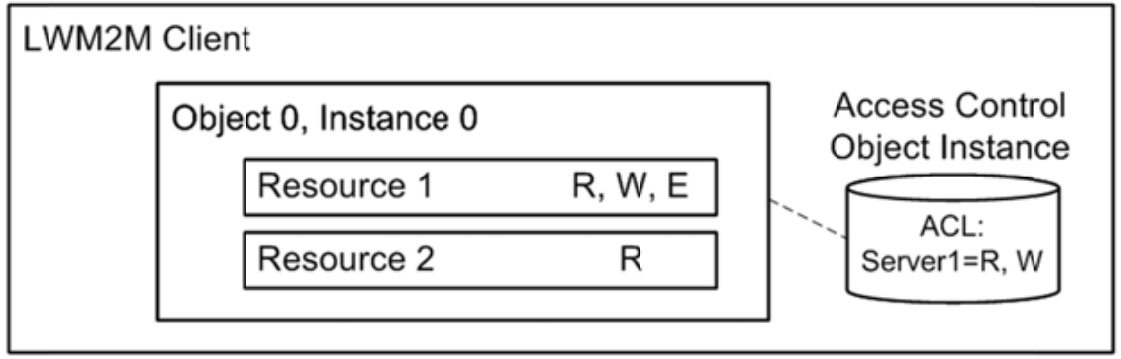
- Observe (Subscribe)
  - Server initiates observation of Object Instance, monitoring changes in the write attributes (minimum/ maximum period, >, <, step, cancel).
- Notify (Publish)
  - Client replies with Notification during observation process. Observation periods are defined in the Observe message.
- Cancel Observation (Unsubscribe)
  - Observation should be cancelled either by specifically sending a cancel observation message or by adding the cancel parameter in the attributes.



IDENTIFIERS AND RESOURCES

› Resource Model

- Everything is a resource. Resources are organized into objects. Objects define a grouping of resources for a purpose (e.g. Firmware upgrade, Error Report, …). Objects must be instantiated before they can be edited. Object Instances should have an Access Control
- Example below: Object Instance 0 has 2 Resources. Resource 1 supports r,w and x operations, Resource 2 only read. The client also has an associated Access Control Object Instance contains an Access Control List (ACL) that supports read an write on that particular object instance. However, writing in Resource 2 is not possible and in Resource 1 he will not be able to execute.



IDENTIFIERS AND RESOURCES: DATA FORMATS

- Identifiers
  - Defined in 6.2 of LWM2M spec, table 16. Provides identifiers for all entities in the protocol: Endpoints, Bootstrap Server, LWM2M Server, Objects, Resources…
  - In particular, Endpoints use URNs (urn:uid, urn:imei, urn:esn, …)
- Data formats for transferring Resource Information
  - Plain Text (if UTF8 r&w), Opaque (if binary r&w), TLV (if multiple instances), JSON.

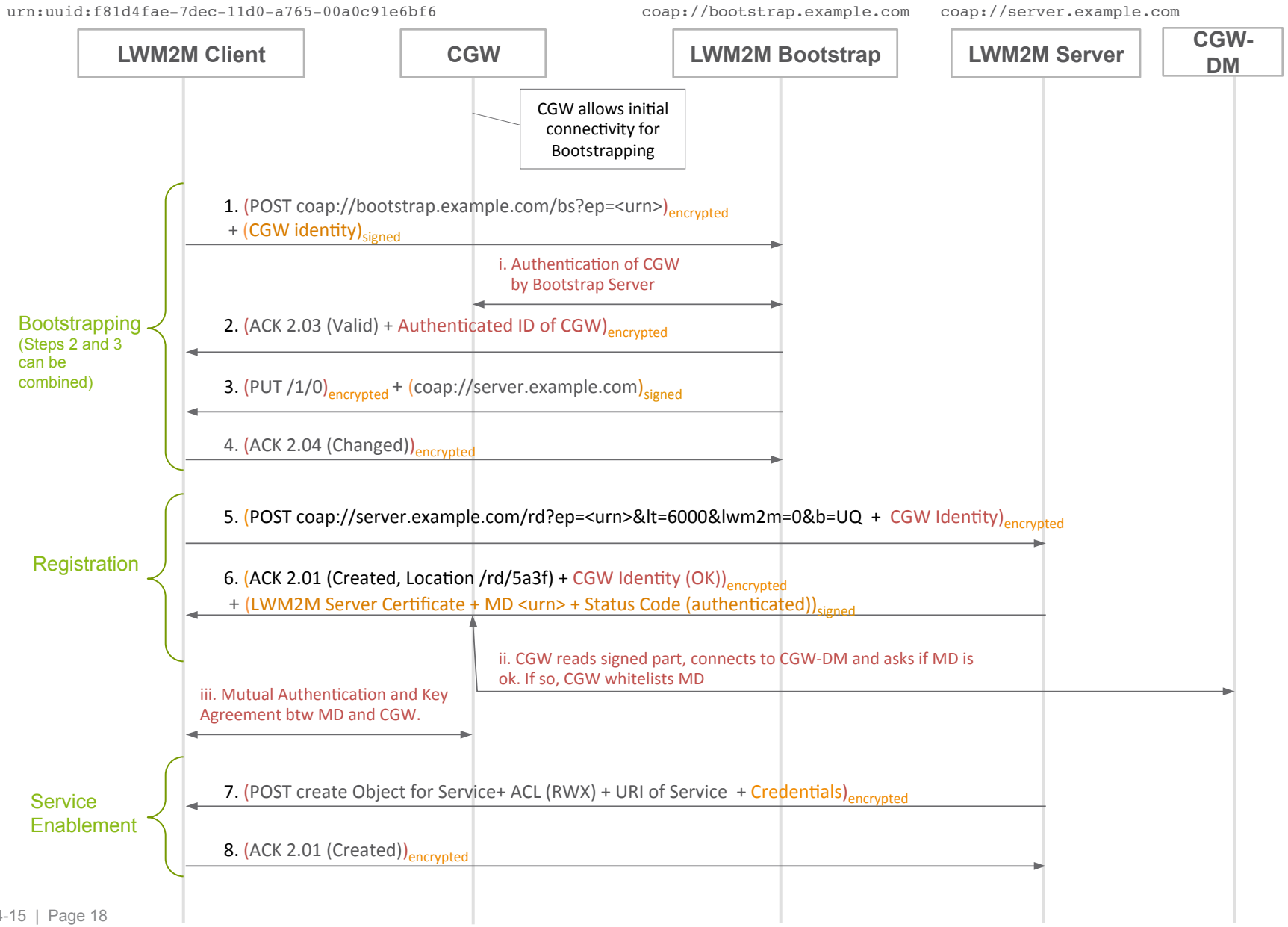
SECURITY

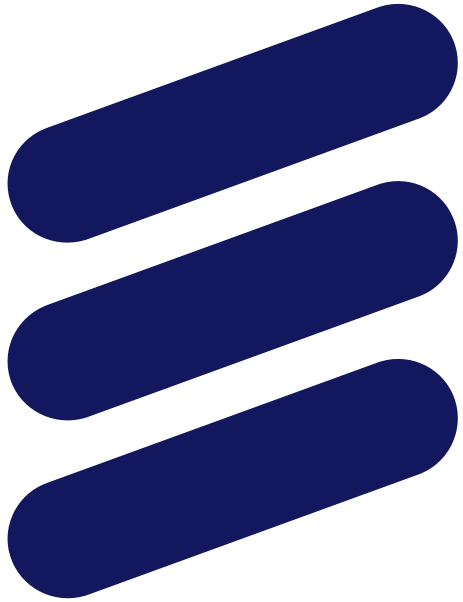
- Based on CoAP.
  - Transport security based on UDP and SMS channel bindings.
  - Authentication of communicating LWM2M entities is required:
    - LWM2M Client MUST authenticate a LWM2M Server prior to exchange of any information.
    - LWM2M Server MUST authenticate a LWM2M Client prior to exchange of any information.
    - LWM2M Client MUST authenticate a LWM2M Bootstrap Server prior to exchange of any information.
    - LWM2M Bootstrap Server MUST authenticate a LWM2M Client prior to exchange of any information.

IMPLEMENTED - TBD

- Introduction
- Bootstrapping
- Registration Interfaces
  - Register, Update, De-register
- Device Management & Service Enablement
  - Read, Discover (RFC6690), Write, Execute, Create, Delete
- Information Reporting
  - Observe, Notify, Cancel Observation
- Identifiers and Resources
- Data Formats
- Security
  - UDP/SMS channel bindings, GBA?
  - Pre-shared keys, PKI, X.509
- Access Control

BOOTSTRAPING





ERICSSON