Enabler Test Specification (Interoperability) for Lightweight M2M

Candidate Version: 1.2.1 - 2024-03-12

Open Mobile Alliance

OMA-ETS-LightweightM2M_INT-V1_2_1-20240312-C

main: 03 Apr 2024 14:20:00 *rev: 13240ce*

Table of Contents

Table of Tables

1. Scope

This document describes in detail available test cases for LightweightM2M as specified in OMA-TS-LightweightM2M-V1_1-20180710-A and OMA-TS-LightweightM2M_Transport-V1_1-20180710-A.

The test cases are split in two categories, conformance and interoperability test cases.

The conformance test cases are aimed to verify the adherence to normative requirements described in the technical specifications.

The interoperability test cases are aimed to verify that implementations of the specifications work satisfactory.

If either conformance or interoperability tests do not exists at the creation of the test specification this part SHOULD be marked not available.

2. References

2.1. Normative References

[3GPP 23.003]	3GPP TS 23.003 "Numbering, addressing and identification"
[CoAP]	Z. Shelby, K. Hartke, C. Bormann, B. Frank, "The Constrained Application Protocol (CoAP)", June 2014, URL:http://www.ietf.org/rfc/rfc7252.txt
[ETSI 102 221]	ETSI TS 102 221 "Smart Cards; UICC-Terminal interface; Physical and logical characteristics",URL:http://www.etsi.org/
[GP SCP02]	GlobalPlatform v2.2.1 - January 2011 - Appendix E: Secure Channel Protocol 02 (SCP 02)
[IOPPROC]	"OMA Interoperability Policy and Process", Version 1.13, Open Mobile Alliance™, OMA-IOP-Process-V1_13, URL:http://www.openmobilealliance.org/
[LwM2M-AD]	"Lightweight Machine to Machine Architecture", Open Mobile Alliance™, OMA-AD-LightweightM2M-V1_0, URL:http://www.openmobilealliance.org/
[LwM2M-CORE]	"Lightweight Machine to Machine Technical Specification: Core Layer", Open Mobile Alliance™, OMA-TS-LightweightM2M_Core-V1_1, URL:http://www.openmobilealliance.org/
[LwM2M-TRANSPORT]	"Lightweight Machine to Machine Technical Specification: Transport Layer", OMA-TS-LightweightM2M_Transport-V1_1, URL:http://www.openmobilealliance.org/
[OBSERVE]	K. Hartke, "Observing Resources in the Constrained Application Protocol (CoAP)", September 2015, URL:https://www.ietf.org/rfc/rfc7641.txt
[PKCS\#15]	PKCS \#15 v1.1: Cryptographic Token Information Syntax Standard", RSA Laboratories, June 6, 2000, URL:ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-15/pkcs-15v1_1.pdf
[BCP 14]	"Key words for use in RFCs to Indicate Requirement Levels", B. Leiba, May 2017, URL:https://www.rfc-editor.org/info/bcp14
[RFC2119]	"Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997, URL:https://www.rfc-editor.org/rfc/rfc2119.txt
[RFC8174]	"Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", B. Leiba, May 2017, URL:https://www.rfc-editor.org/rfc/rfc8174.txt
[RFC2234]	"Augmented BNF for Syntax Specifications: ABNF". D. Crocker, Ed., P. Overell. November 1997, ,URL:http://www.ietf.org/rfc/rfc2234.txt
[RFC4122]	P. Leach, M. Mealling, R. Salz, "A Universally Unique Identifier (UUID) URN Namespace", July 2005,URL:http://www.ietf.org/rfc/rfc4122.txt
[RFC5246]	T. Dierks, E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", August 2008, URL:https://www.ietf.org/rfc/rfc5246.txt
[RFC5289]	E. Rescorla, "TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)", August 2008, URL:https://www.ietf.org/rfc/rfc5289.txt
[RFC5487]	M. Badra, "Pre-Shared Key Cipher Suites for TLS with SHA-256/384 and AES Galois Counter Mode", March 2009, URL:https://www.ietf.org/rfc/rfc5487.txt
[RFC6347]	E. Rescorla, N. Modadugu, "Datagram Transport Layer Security Version 1.2", January 2012, URL:https://www.ietf.org/rfc/rfc6347.txt
[RFC6655]	D. McGrew, D. Bailey, "AES-CCM Cipher Suites for TLS", July 2012,URL:https://www.ietf.org/rfc/rfc6655.txt

[RFC6690]	Z. Shelby, "Constrained RESTful Environments (CoRE) Link Format", August 2012,URL:https://www.ietf.org/rfc/rfc6690.txt

Table: 2.1.-1 Normative References

2.2. Informative References

3. Terminology and Conventions

3.1. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

All sections and appendixes, except "Scope", are normative, unless they are explicitly indicated to be informative.

The following numbering scheme is used:

xxx-y.z-con-number where:
xxx Name of enabler, e.g. MMS or Browsing
y.z Version of enabler release, e.g. 1.2 or 1.2.1
'con' Indicating this test is a conformance test case
number Leap number for the test case

Or

xxx-y.z-int-number where:
xxx Name of enabler, e.g. MMS or Browsing
y.z Version of enabler release, e.g. 1.2 or 1.2.1
'int' Indicating this test is a interoperability test case
number Leap number for the test case

3.2. Definitions

API	Application Programming Interface
APN	Access Point Name
CoAP	Constrained Application Protocol
CON	Conformance
DM	Device Management
GDSP	Global Data Service Platform
GUI	Graphical User Interface
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
IOP	Interoperability
LwM2M	Lightweight Machine to Machine (refers to this OMA enabler)
M2M	Machine to Machine
MSISDN	Mobile Station International Subscriber Directory Number

OMA	Open Mobile Alliance
OpCo	Operating Company
OS	Operating System
SIM	Subscriber Interface Module
UI	User Interface

Table: 3.2.–1 Definitions

3.3. Abbreviations

4. Introduction

The purpose of this document is to provide test cases for LightweightM2M Enabler Release V1.0.

The implementation of some features is optional for the Clients and/or the Servers in the LightweightM2M Enabler. The tests associated with these optional features are marked as "(Includes Optional Features)" in the test specification.

The following items on an overall level are needed to adequately test the LwM2M enabler:

- A LwM2M Server
- A LwM2M client e.g. embedded in a M2M device or module connected via UDP and SMS with the LwM2M Server

The LwM2M enabler tests are carried out using the LwM2M protocol and objects, and using the underlying protocols such as [CoAP].

The four data formats used in that document, namely Plain Text, Opaque, TLV and JSON data formats are respectively associated to the text/plain, application/octet-stream, application/vnd.oma.lwm2m+tlv, application/vnd.oma.lwm2m+json Media Types referred by LwM2M TS 1.0 and registered in IANA.

5. LightweightM2M Conformance Test Cases

None.

6. LightweightM2M Interoperability Test Cases

6.1. LwM2M Interfaces Test Cases [0-99]

6.1.1. Bootstrap Interface: [0-99]

6.1.1.1. LightweightM2M-1.1-int-0 -- Client Initiated Bootstrap

Test Case Id

LightweightM2M-1.1-int-0

Test Object

Client and Server

Test Case Description

Test the Client capability to connect the Bootstrap Server according to the Client Initiated Bootstrap Mode

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.13 as defined in Annex C
- The Client is supporting the Client Initiated Bootstrap Mode
- The Client doesn't contain any Instance of LwM2M Server Object ID:1 (minimum condition to trigger a Bootstrap request message when the device is on)

Test Procedure

The Client requests the Bootstrap Server to setup a configuration enabling the Client to contact a LwM2M Server.

Normal flow:

1. Without Instance of Server Object, the Client performs a BOOTSTRAP-REQUEST (CoAP POST /bs?ep{Endpoint Client Name}) in using the Resource values of the Instance 1 of Security Object ID:0 to contact the Bootstrap Server
2. The Bootstrap Server updates the Configuration in the Client (C13+C1) in performing two BOOTSTRAP-WRITE operations (CoAP PUT /0, CoAP PUT /1) in using the set of values defined in 0-SetOfValue
3. The Bootstrap Server performs a BOOTSTRAP-DISCOVER operation (CoAP GET Accept:40 /) to verify the Client setup
4. The Bootstrap Server performs a BOOTSTRAP-FINISH operation (CoAP POST /bs) to end-up that BS phase

Pass-Criteria

1. In test step 1., the Bootstrap Server received a Success Message ("2.04" Changed) related to the BOOTSTRAP-REQUEST of the Client
2. In test step 2., Client received WRITE operation(s) to update the initial configuration with the values contains in 0-SetOfValue
3. In test step 2., Server received Success ("2.04" Changed) message(s) from Server related to WRITE operation(s)
4. In test step 3., the Bootstrap Server received the Success message ("2.05" Content) along with the payload related to the BOOTSTRAP-DISCOVER request and containing :\

   lwm2m="1.1",</0/0>;ssid=1,</0/1>, </1/0>;ssid=1,</3/0>
5. In test step 4., the Bootstrap Server received the Success message ("2.04" Changed) : no inconsistency detected

0-SetOfValues definition

lwm2m+json version

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","bs":"U"}]

6.1.1.2. LightweightM2M-1.1-int-1 -- Client Initiated Bootstrap Full (PSK)

Test Case Id

LightweightM2M-1.1-int-1

Test Object

Client and Server

Test Case Description

Test the Client capability to connect the Bootstrap Server according to the Client Initiated Bootstrap Mode followed by

registering to the LwM2M Server afterwards. In this test case the Bootstrap Server provisions the LwM2M Client for PSK-based authentication.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.13 as defined in Annex C
- The Client is supporting the Client Initiated Bootstrap Mode
- The Client doesn't contain any Instance of LwM2M Server Object ID:1 (minimum condition to trigger a Bootstrap request message when the device is on)
- The Client is capable of using PSK-based DTLS credentials

Test Procedure

The Client requests the Bootstrap Server to setup a configuration enabling the Client to contact a LwM2M Server.

Normal flow:

1. Without Instance of Server Object, the Client performs a BOOTSTRAP-REQUEST (CoAP POST /bs?ep{Endpoint Client Name}) in using the Resource values of the Instance 1 of Security Object ID:0 to contact the Bootstrap Server
2. The Bootstrap Server uploads the Configuration C.1 in the Client in performing two BOOTSTRAP-WRITE (CoAP PUT /0, CoAP PUT /1) in using the set of values defined in 1-SetOfValues Object Device ID:3,is automatically created and filled-up by the Client)
3. The Bootstrap Server performs a BOOTSTRAP-DISCOVER operation (CoAP GET Accept:40 /) to verify the Client setup
4. The Bootstrap Server performs a BOOTSTRAP-FINISH operation (CoAP POST /bs) to end-up that BS phase
5. The LwM2M Client connects to the provisioned LwM2M server using the PSK-based credentials, as described in LightweightM2M-1.1-int-401

Pass-Criteria

1. In test step 1., the Bootstrap Server received a Success Message ("2.04" Changed) related to the BOOTSTRAP-REQUEST of the Client
2. In test step 2., Client received WRITE operation(s) to setup the C.1 configuration (1-SetOfValues)
3. In test step 2., Server received Success ("2.04" Changed) message(s) from Server related to WRITE operation(s)
4. In test step 3., the Bootstrap Server received the Success message ("2.05" Content) along with the payload related to the BOOTSTRAP-DISCOVER request and containing :\

   lwm2m="1.1",</0/0>;ssid=1,</0/1>, </1/0>;ssid=1,</3/0>

5. In test step 4., the Bootstrap Server received the Success message ("2.04" Changed) : no inconsistency detected
6. The test case in LightweightM2M-1.1-int-401 executes successfully.

1-SetOfValues definition

lwm2m+json version

[{"bn":"/", "n":"0/0/0","vs": <LwM2M Server URI>},
{"n":"0/0/1","vb":false},
{"n":"0/0/2","v":0},
{"n":"0/0/3","vs":<PSK Identity>}, // opaque in Base64
{"n":"0/0/4","vs": <n/a> },
{"n":"0/0/5","vs": <Secret Key>}, // opaque in Base64
{"n":"0/0/10","v":1},
{"n":"1/0/0","v":1},
{"n":"1/0/1","v":86400},
{"n":"1/0/6","vb":false},
{"n":"1/0/7","vs":"U"}]

6.1.1.3. LightweightM2M-1.1-int-2 -- Client Initiated Bootstrap Full (Cert)

Test Case Id

LightweightM2M-1.1-int-2

Test Object

Client and Server

Test Case Description

Test the Client capability to connect the Bootstrap Server according to the Client Initiated Bootstrap Mode followed by registering to the LwM2M Server afterwards. In this test case the Bootstrap Server provisions the LwM2M Client for certificate-based authentication.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.13 as defined in Annex C
- The Client is supporting the Client Initiated Bootstrap Mode
- The Client doesn't contain any Instance of LwM2M Server Object ID:1 (minimum condition to trigger a Bootstrap request message when the device is on)
- The Client is capable of using certificate-based DTLS credentials

Test Procedure

The Client requests the Bootstrap Server to setup a configuration enabling the Client to contact a LwM2M Server.

Normal flow:

1. Without Instance of Server Object, the Client performs a BOOTSTRAP-REQUEST (CoAP POST /bs?ep{Endpoint Client Name}) in using the Resource values of the Instance 1 of Security Object ID:0 to contact the Bootstrap Server
2. The Bootstrap Server uploads the Configuration C.1 in the Client in performing two BOOTSTRAP-WRITE (CoAP PUT /0, CoAP PUT /1) in using the set of values defined in 2-SetOfValues (Object Device ID:3,is automatically created and filled-up by the Client)
3. The Bootstrap Server performs a BOOTSTRAP-DISCOVER operation (CoAP GET Accept:40 /) to verify the Client setup
4. The Bootstrap Server performs a BOOTSTRAP-FINISH operation (CoAP POST /bs) to end-up that BS phase
5. The LwM2M Client connects to the provisioned LwM2M server using the certificate-based credentials, as described in LightweightM2M-1.1-int-402

Pass-Criteria

1. In test step 1., the Bootstrap Server received a Success Message ("2.04" Changed) related to the BOOTSTRAP-REQUEST of the Client
2. In test step 2., Client received WRITE operation(s) to setup the C.1 configuration (2-SetOfValues)
3. In test step 2., Server received Success ("2.04" Changed) message(s) from Server related to WRITE operation(s)
4. In test step 3., the Bootstrap Server received the Success message ("2.05" Content) along with the payload related to the BOOTSTRAP-DISCOVER request and containing:

lwm2m="1.1",</0/0>;ssid=1,</0/1>, </1/0>;ssid=1,</3/0>

5. In test step 4., the Bootstrap Server received the Success message ("2.04" Changed) : no inconsistency detected
6. The test case in LightweightM2M-1.1-int-402 executes successfully.

2-SetOfValues definition

lwm2m+json version

[{"bn":"/", "n":"0/0/0","vs":coaps://server.example.com },

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":2},

{"n":"0/0/3","vs":<Client Certificate >}, // Contains the client.crt

{"n":"0/0/4","vs": <Server Certificate> }, // Contains the server.crt

{"n":"0/0/5","vs": <Client Private Key>}, // Contains the client.key

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"}]

6.1.1.4. LightweightM2M-1.1-int-3 -- Simple Bootstrap from Smartcard

Test Case Id

LightweightM2M-1.1-int-3

Test Object

Client and Server

Test Case Description

Test the Client capability to connect the expected LwM2M Server according to the Bootstrap Information uploaded from the Smartcard (Bootstrap from Smartcard)

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the "Bootstrap from Smartcard" mode as defined in LwM2M 1.1: no Smartcard with Bootstrap Information is inserted
- The Client is configured with the minimum Configuration C.1 as defined in Annex C with LwM2M Object Server URI "X" & Lifetime=86400s
- The Client is connected to the LwM2M Server X
- A Smartcard is prepared containing a Bootstrap Information file as specified in Annex G and H of the TS LwM2M 1.0.2; this Bootstrap Information contains a Server Account different from the C.1 one (LwM2M Server Object URI "Y" with "Y" different of "X" and Lifetime of 43200s - ID :"/1/0/1")

Test Procedure

The Device is powered--off and the prepared Smartcard (pre-conditions) is inserted in the Device. After power-on, the Bootstrap procedure from Smartcard occurs and the Client registers to LwM2M Server "Y".

Normal flow:

1. The LwM2M Client is connected to the LwM2M Server "X"
2. The LwM2M Server retrieves (READ /1/0/1) the Server Object lifetime of the LwM2M Client
3. The Device is powered-off, the prepared Smartcard is inserted in the Device, then the Device is powered-on again.
4. The Client registers to the LwM2M Server "Y" as specified in the Smartcard Bootstrap Information file.
5. The LwM2M Server "Y" verifies (READ /1/0/1) that the Server Object lifetime contains the expected value (as specified in the Smartcard Bootstrap Information file)

Pass-Criteria

1. In test step 1., Initial Registration for Server "X" has passed (pass criteria A&B of Test Case int-101)
2. In test step 2, the READ operation on the Resource "Lifetime" of the Server Object (READ /1/0/1) of the LwM2M Client provides the value given by the C.1 configuration (86400) along with the success message ("2.05" Content)
3. In test step 4, Initial Registration for Server "Y" has passed (pass criteria A&B of Test Case int-101 but in using Configuration set by the Bootstrap from Smartcard)
4. In test step 5, the READ operation on the Resource "Lifetime" of the Server Object (READ /1/0/1) of the LwM2M Client provides the value given by the new configuration (43200) along with the success message ("2.05" Content)

6.1.1.5. LightweightM2M-1.1-int-4 - Bootstrap Delete

Test Case Id

LightweightM2M-1.1-int-4

Test Object

Client and Server

Test Case Description

Test, that both the Client and Server supports Bootstrap-DELETE operation and have the capability to perform removal of all previous bootstrap configurations

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.13 as defined in Annex C
- The Client supports the Client Initiated Bootstrap Mode

Test Procedure

1. LwM2M Client performs a BOOTSTRAP-REQUEST (CoAP POST /bs?ep={Endpoint Client Name}) by using the Resource values of the appropriate Instance of Security Object ID:0 to contact the Bootstrap Server
2. The Bootstrap Server continues with standard 'Client Initiated Bootstrap' procedure, as in LightweightM2M-1.1-int-0 test case (Bootstrap-Discover, Bootstrap-Writes and Bootstrap).
3. The Bootstrap Server sends a Bootstrap-Discover (CoAP GET Accept:40) together with Bootstrap-Read (CoAP GET) on LwM2M Server Object /1
4. The Bootstrap Server sends a Bootstrap-DELETE (CoAP DELETE) on Objects "/0" and "/1" to cleanse the previous bootstrap configuration
5. The Bootstrap Server sends a Bootstrap-Discover (CoAP GET Accept:40) together with Bootstrap-Read (CoAP GET) on LwM2M Server Object /1
6. Test clean-up: The Bootstrap Server continues with standard 'Client Initiated Bootstrap' procedure, as in LightweightM2M-1.1-int-0 test case (Bootstrap-Discover, Bootstrap-Writes and Bootstrap-Finish).

Pass-Criteria

1. The Bootstrap Server received a Success Message ("2.04" Changed) related to the BOOTSTRAP-REQUEST of the Client
2. Client successfully accepts consecutive messages (2.04 Changed) and accepts Bootstrap information.
3. The Bootstrap Server received the Success message (CoAP 2.05 Content) along with the payload related to the BOOTSTRAP-DISCOVER and Bootstrap-READ request containing a newly created Object Instance of LwM2M Server (/1/)
4. Client received Bootstrap-DELETE operation "/0" and "/1" target and replied with "2.02 Deleted". LwM2M Bootstrap-Server Account (appropriate Instance of Security Object) is not affected by any Delete operation.
5. The Bootstrap Server received the Success message (CoAP 2.05 Content) along with the payload related to the Bootstrap-DISCOVER and Bootstrap-READ. Client's replies indicate that Object Instance of LwM2M Server was in fact deleted
6. Client successfully accepts consecutive messages (2.04 Changed) and accepts Bootstrap information. Bootstrap procedure ends with Success reply to Bootstrap-FINISH. Client proceeds with registration to LwM2M Server.

6.1.1.6. LightweightM2M-1.1-int-5 - Server Initiated Bootstrap

Test Case Id

LightweightM2M-1.1-int-5

Test Object

Client and Server

Test Case Description

Test the Client and Server capability to perform Server Initiated Bootstrap (Server triggers Client Initiated Bootstrap)

Tool

n/a

Test code

n/a

Preconditions

- Connection between a LwM2M Client and a LwM2M Server MUST already exist (Client is registered)

Test Procedure

1. Server executes a 'Bootstrap-Request Trigger' resource, by sending CoAP POST on /1/x/9 (where x is a LwM2M Server Object Instance)

Pass-Criteria

1. In test step 1. Client accepts the command and replies with 2.04 Changed.
2. Then, Devices send a Bootstrap-REQUEST (POST on /bs?ep={endpointName}) as it would perform a standard Client Initiated Bootstrap (LightweightM2M-1.1-int-0 test case)
3. Client eventually receives Bootstrap Information and whole procedure ends with Bootstrap-FINISH, as in *LightweightM2M-1.1-int-0* test case

6.1.1.7. LightweightM2M-1.1-int-6 - Bootstrap Sequence

Test Case Id

LightweightM2M-1.1-int-6

Test Object

Client

Test Case Description

Test if Client respects Bootstrap priority Sequence.

The LwM2M Client MUST respect step by step the procedural sequence when attempting to bootstrap a LwM2M Device, although it's required to support only one bootstrap procedure mode. Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.13 as defined in Annex C
- The Client supports at least 2 Bootstrap modes

Test Procedure

1. Make the device start bootstrapping (i.e. enable the Device.)

Pass-Criteria

1. If the LwM2M Device has Smartcard, the LwM2M Client tries to obtain Bootstrap Information from the Smartcard using the Bootstrap from Smartcard mode
2. If the LwM2M Client is not configured using the Bootstrap from Smartcard mode, the LwM2M Client tries to obtain the Bootstrap Information by using Factory Bootstrap mode.
3. If the LwM2M Client has any LwM2M Server Object Instances from the previous steps, the LwM2M Client tries to register to the LwM2M Server(s) configured in the LwM2M Server Object Instance(s).
4. If the LwM2M Client fails to register to all the LwM2M Servers or the Client doesn't have any LwM2M Server Object Instances, the LwM2M Client performs the Client Initiated Bootstrap.
5. A Server Initiated Bootstrap attempt (e.g. for updating a LwM2M Server Account) remains possible, but only if the LwM2M Client retains the corresponding LwM2M Bootstrap-Server Account.

6.1.1.8. LightweightM2M-1.1-int-7 - Fallback to bootstrap

Test Case Id

LightweightM2M-1.1-int-7

Test Object

Client and Server

Test Case Description

Test if Client respects Bootstrap priority Sequence and correctly fallbacks to Bootstrap when registration fails

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.13 as defined in Annex C
- Client has valid Bootstrap server provisioned

Test Procedure

1. Provision the Client with incorrect (e.g. non-existing) LwM2M Server information, so the Client uses Factory

Bootstrap mode

2. Make the Client start bootstrapping.

Pass-Criteria

1. Client tries to register to non-existing LwM2M Server which ends with failure (LwM2M Server does not exist).
2. When the LwM2M Client fails to register to previous LwM2M Server - the Client performs Client Initiated Bootstrap as described in LightweightM2M-1.1-int-0
3. Client initiated Bootstrap is accepted by a Bootstrap Server. After successful bootstrap procedure the LwM2M Client correctly registers to LwM2M Server (as in *LightweightM2M-1.1-int-401*)

6.1.1.9. LightweightM2M-1.1-int-8 - Bootstrap Read

Test Case Id

LightweightM2M-1.1-int-8

Test Object

Client and Server

Test Case Description

Test if Client and Server have capability of performing Bootstrap-READS

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.13 as defined in Annex C
- Client is capable of running Client Initiated Bootstrap

Test Procedure

1. LwM2M Client performs a BOOTSTRAP-REQUEST (CoAP POST /bs?ep{Endpoint Client Name}) by using the Resource values of the appropriate Instance of Security Object ID:0 to contact the Bootstrap Server
2. The Bootstrap Server uploads the Configuration C.1 in the Client in performing two BOOTSTRAP-WRITE (CoAP PUT /0, CoAP PUT /1) by using the set of values defined in 8-SetOfValues (Object Device ID:3,is automatically created and filled-up by the Client)
3. The Bootstrap Server initiates Access Control (/2) Object configuration using Bootstrap-WRITE with the following values:

/2/0/0 = 3

/2/0/1 = 0

/2/0/2/0 = 31

/2/0/3 = 65535

4. The Bootstrap Server performs Bootstrap-READs on LwM2M Server Object (/1) and on Access Control Object (/2) to verify correctness of newly uploaded Bootstrap Information
5. The Bootstrap Server performs a BOOTSTRAP-FINISH operation (CoAP POST /bs) to end-up that BS phase

Pass-Criteria

1. In test step 1., the Bootstrap Server sent a Success Message ("2.04" Changed) related to the BOOTSTRAP-REQUEST of the Client
2. In test step 2., Client received Bootstrap-WRITE (CoAP PUT) operation(s) to setup the C.1 configuration (8-SetOfValues) and replied with 2.04 Created
3. In test step 3., Client received Bootstrap-WRITE (CoAP PUT) operation(s) to setup Access Control Object and replied with 2.04 Created
4. In test step 4., Client received Bootstrap-READ (CoAP GET) operations to read on LwM2M Server (/1) and Access Control (/2) Objects. Client replied with 2.05 Content
5. Procedure ends with Success. Device replied with 2.04 Changed response for Bootstrap-Finish request (CoAP POST /bs).

8-SetOfValues definition

lwm2m+json version

[{"bn":"/", "n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vs":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vs": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"}]

6.1.1.10. LightweightM2M-1.1-int-9 - Bootstrap and Configuration Consistency

Test Case Id

LightweightM2M-1.1-int-9

Test Object

Client and Server

Test Case Description

Test if Client detects inconsistency in received Bootstrap information (i.e. mandatory LwM2M Server resources are missing)

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.13 as defined in Annex C
- Client is capable of running Client Initiated Bootstrap

Test Procedure

1. LwM2M Client performs a BOOTSTRAP-REQUEST (CoAP POST /bs?ep{Endpoint Client Name}) by using the Resource values of the appropriate Instance of Security Object ID:0 to contact the Bootstrap Server
2. The Bootstrap Server uploads only the value of non-mandatory resource *Disable Timeout of LwM2M Server Object* (/1) and sets it to 86400 with CoAP PUT on /1/x/5, where x is Object Instance chosen by Server.
3. The Bootstrap Server performs a BOOTSTRAP-FINISH operation (CoAP POST /bs) to end-up BS phase

Pass-Criteria

1. In test step 1., the Bootstrap Server sent a Success Message ("2.04" Changed) related to the BOOTSTRAP-REQUEST of the Client
2. In test step 2., Client received Bootstrap-WRITE (CoAP PUT) operation(s) with specified Resource value and replied with 2.04 Created
3. Procedure finished with error - Client replied with 4.06 Not Acceptable CoAP response for Bootstrap-Finish request (CoAP POST /bs). Device did not accept the Bootstrap Information, because mandatory LwM2M Server Account information is missing (LwM2M Security Object and other resources from LwM2M Server Object)

6.1.1.11. LightweightM2M-1.1-int-10 -- Client Initiated Bootstrap Full (EST)

Test Case Id

LightweightM2M-1.1-int-10

Test Object

LwM2M Client, LwM2M Bootstrap-Server and LwM2M Server

Test Case Description

Test the LwM2M Client capability to connect the LwM2M Bootstrap-Server according to the Client Initiated Bootstrap procedure followed by registering to the LwM2M Server afterwards. In this test case the LwM2M Client uses EST to provision a certificate to the LwM2M Client.

Tool

n/a

Test code

n/a

Preconditions

- The LwM2M Client supports the Configuration C.21, as defined in Annex C, at the start of the test. The LwM2M Client is capable of using certificate-based credentials with EST.
- The LwM2M Client is supporting the Client Initiated Bootstrap Mode.
- The LwM2M Client does not contain any Instance of LwM2M Server Object ID:1 (minimum condition to trigger a Bootstrap request message when the device is on).

Test Procedure

The LwM2MClient requests the LwM2M Bootstrap-Server to setup a configuration enabling the LwM2M Client to contact the LwM2M Server.

Normal flow:

1. The LwM2M Client connects to the LwM2M Bootstrap-Server using EST with the information configured in the LwM2M Security Object. This requires the LwM2M Client to create a key pair locally on the device and to submit the public key via a Certificate Signing Request (CSR) to the LwM2M Bootstrap-Server.
2. The LwM2M Bootstrap-Server returns a certificate after authenticating and verifying the LwM2M Client.
3. Without an Instance of the LwM2M Server Object configured, the LwM2M Client performs a BOOTSTRAP-REQUEST (CoAP POST /bs?ep{Endpoint Client Name}).
4. The LwM2M Bootstrap Server performs a BOOTSTRAP-WRITE operation to setup the configuration (0-SetOfValue).
5. The LwM2M Server receives a Success ("2.04" Changed) message from the LwM2M Client.
6. The LwM2M Bootstrap-Server performs a BOOTSTRAP-FINISH operation to finalize the bootstrap phase.
7. The configured information kicks off the registration with the LwM2M Server. The LwM2M Client connects to the provisioned LwM2M Server and successfully registers.

Pass-Criteria

1. The LwM2M Bootstrap-Server received a Success Message ("2.04" Changed) related to the BOOTSTRAP-REQUEST of the LwM2M Client
2. The LwM2M Client received WRITE operation(s) to setup the configuration.
3. The LwM2M Bootstrap-Server receives a Success ("2.04" Changed) message related to the WRITE operation.
4. The EST exchange runs successfully and the LwM2M Client obtains a certificate for use with the LwM2M Server.
5. The LwM2M Client has obtained the necessary information to successfully execute a registration procedure with the configured LwM2M Server.

0-SetOfValues definition

lwm2m+json version

[{"bn":"/", "n":"0/0/0","vs":coaps://server.example.com },

{"n":"0/1/1","vb":false},

{"n":"0/1/2","v":2},

{"n":"0/1/3","vs": ""}, // empty (use from EST)

{"n":"0/1/4","vs": <LwM2M Server Certificate> }, // Contains the LwM2M Server Public Key

{"n":"0/1/5","vs": ""}, // empty (use from EST)

{"n":"0/1/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"}]

6.1.2. LightweightM2M-1.2-int-11 -- Client Initiated Bootstrap (OSCORE Security)

Test Case Id

LightweightM2M-1.2-int-11

Test Object

Client and Server

Test Case Description

Test the Client capability to connect the Bootstrap Server according to the Client Initiated Bootstrap Mode with OSCORE security and get configured with OSCORE security objects for server registration

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.23 as defined in Annex C
- The Client is supporting the Client Initiated Bootstrap Mode
- The Client doesn't contain any Instance of LwM2M Server Object ID:1 (minimum condition to trigger a Bootstrap request message when the device is on)

Test Procedure

The Client requests the Bootstrap Server to setup a configuration enabling the Client to contact a LwM2M Server.

Normal flow:

1. Without Instance of Server Object, the Client performs a BOOTSTRAP-REQUEST (CoAP POST /bs?ep{Endpoint Client Name}) in using the Resource values of the Instances 0 of Security Object ID:0 and OSCORE Object ID:21 as per C.23 configuration to contact the Bootstrap Server
2. The Bootstrap Server updates the Configuration in the Client (C23+C24) in performing three BOOTSTRAP-WRITE operations (CoAP PUT /0, CoAP PUT /21 and CoAP PUT /1) in using the set of values defined in 0-SetOfValue
3. The Bootstrap Server performs a BOOTSTRAP-DISCOVER operation (CoAP GET Accept:40 /) to verify the Client setup
4. The Bootstrap Server performs a BOOTSTRAP-FINISH operation (CoAP POST /bs) to end-up the BS phase

Pass-Criteria

1. In test step 1., the Bootstrap Server replies with a Success Message ("2.04" Changed) related to the BOOTSTRAPREQUEST of the Client
2. In test step 2., Client received WRITE operation(s) to update the initial configuration with the values contains in 0-SetOfValue
3. In test step 2., Server received Success ("2.04" Changed) message(s) from Client related to WRITE operation(s)
4. In test step 3., the Bootstrap Server received the Success message ("2.05" Content) along with the payload related to the BOOTSTRAP-DISCOVER request and containing: lwm2m="1.1",</0/0>,</0/1>; ssid=1; uri="coap://server_1.example.com",</1/0>;ssid=1,</3/0>,</21/0>,</21/1>;ssid=1
5. In test step 4., the Bootstrap Server received the Success message ("2.04" Changed) : no inconsistency detected

0-SetOfValues definition

lwm2m+json version

[{"bn":"/0/1/"},

{"n":0","vs": <LwM2M Server URI>},

{"n":"1","vb":false},

{"n":"2","v":3},

{"n":"3","vd":<n/a>},

{"n":"4","vs": <n/a>},

{"n":"5","vd": <n/a>},

{"n":"10","v":1},

{"n":"17","v":"/21/1"},

{"bn":"/21/1/"},

{"n":"0","v":<Master Salt Key>}, // opaque in Base64

{"n":"1","v":<Sender ID>},

{"n":"2","v":<Receiver ID>},

{"bn":"/1/1/","n":"0","v":1},

{"n":"1","v":86400},

{"n":"6","vb":false},

{"n":"7","vs":"U"}]

6.1.2.1. LightweightM2M-1.2-int-12 -- Client bootstrap through use of Bootstrap-Pack Request to the BS

Test Case Id

LightweightM2M-1.2-int-12

Test Object

Client and Server

Test Case Description

Test the Client capability to connect the Bootstrap Server according to the Client Initiated Bootstrap Mode using Bootstrap-Pack-Request Operation followed by registering to the LwM2M Server afterwards. In this test case the Bootstrap Server provisions the LwM2M Client for PSK-based authentication.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.13 as defined in Annex C
- The Client and server are both supporting Bootstrap-Pack

- The Client doesn't contain any Instance of LwM2M Server Object ID:1 (minimum condition to trigger a Bootstrap request message when the device is on)
- The Client is capable of using PSK-based DTLS credentials

Test Procedure

The Client requests the Bootstrap Server to setup a configuration enabling the Client to contact a LwM2M Server.

Normal flow:

1. Without Instance of Server Object, the Client performs a BOOTSTRAP-PACK-REQUEST (CoAP GET /bspack?ep{Endpoint Client Name} &acc={LwM2M Bootstrap- Server Account Object Instances}) in using the Resource values of the Instance 1 of Security Object ID:0 to contact the Bootstrap Server
2. The Bootstrap Server responds with instances of Objects 0 and 1 using the set of values defined in 0-SetOfValue in its 2.05 COAP response. Object Device ID:3,is automatically created and filled-up by the Client to finish with the configuration C.1.
3. **The LwM2M Client connects to the provisioned LwM2M server using the PSK-based credentials, as described in LightweightM2M-1.1-int-401**

Pass-Criteria

1. In test step 1., the Bootstrap Server received a Success Message ("2.04" Changed) related to the BOOTSTRAP-REQUEST of the Client
2. In test step 2., Client received BOOTSTRAP-PACK Response to setup the C.1 configuration (0-SetOfValue)
3. The test case in LightweightM2M-1.1-int-401 executes successfully.

0-SetOfValues definition

lwm2m+json version

[{"bn":"/", "n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vs":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vs": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"}]

6.1.2.2. LightweightM2M-1.2-int-19 -- Client Initiated Bootstrap over MQTT

Test Case Id

LightweightM2M-1.2-int-19

Test Object

Client and Server

Test Case Description

Test the Client capability to connect the Bootstrap Server over MQTT according to the Client Initiated Bootstrap Mode to receive LwM2M Server Account information that it can use for subsequent LwM2M registration over MQTT. In this test case the Bootstrap Server provisions the LwM2M Client with PSK-based authentication for MQTT Server/Broker and COSE information to secure message exchanges with LwM2M Server that it will register with.

Tool

n/a

Test code

n/a

Preconditions

- Bootstrap Server Contact Information as per Configuration C.25 is present on the client
- The Client supports the minimum Configuration C.26 as defined in Annex C
- The Client is supporting the Client Initiated Bootstrap Mode
- The Client doesn't contain any Instance of LwM2M Server Object ID:1 (minimum condition to trigger a Bootstrap request message when the device is on)
- The Client supports MQTT Transport
- The Client supports COSE
- The Client is capable of using PSK-based DTLS credentials
- Bootstrap Server supports MQTT transport and has subscribed to "+/lwm2m/bs/+"

Test Procedure

The Client requests the Bootstrap Server to setup a configuration enabling the Client to contact a LwM2M Server.

Normal flow:

1. Without an Instance of Server Object, the Client uses the Resource values of the Instances 0 of Security Object ID:0 as well as Client Identifier and flags that are configured in object instance /23/0 to establishes a secure MQTT connection to the MQTT Server/Broker.
2. Client will subscribe to "+/lwm2m/bs/"<Client Endppoint Name>

3. Client will send Bootstrap Request to the bootstrap server by publishing to "lwm2m/bs/"<Client Endppoint Name> with the payload of
    - { operation => 0, token => uint, pct => 60 }
4. The Bootstrap Server upload Configuration C.26 to the Client for object IDs 0, 1, 23 and 24 by publishing 4 BOOTSTRAP-WRITE operations to "+/lwm2m/bs/"<Client Endppoint Name> topic using the set of values defined in 0-SetOfValue (Object Device ID:3,is automatically created and filled-up by the Client)
5. The Bootstrap Server performs a BOOTSTRAP-DISCOVER operation to verify the Client setup
    - { operation => 4, token => uint, uri => "/" }
6. The Bootstrap Server performs a BOOTSTRAP-FINISH operation to end Bootstrap phase

Pass-Criteria

1. In test step 1., client receives MQTT CONNACK to confirm establishment of MQTT connection
2. In test step 2., client receives SUBACK to confirm its successful subscription to specified topic
3. In test step 3., client receives PUBACK
4. In test step 4., Client receives WRITE operation(s) to setup the C.26 configuration (0-SetOfValue)
5. In test step 4., Server receives Success ("2.04" Changed) message(s) from client related to WRITE operation(s)
6. In test step 5., the Bootstrap Server received the Success message ("2.05" Content) along with the payload related to the BOOTSTRAP-DISCOVER request and containing :

   \lwm2m="1.2",</0/0>,</0/1>; ssid=1; uri="mqtt://server_1.example.com",</1/0>;ssid=1,</3/0>,</23/0>, </23/1>;ssid=1,</24/0>,</24/1>;ssid=1
7. In test step 6., the Bootstrap Server received the Success message ("2.04" Changed) : no inconsistency detected

0-SetOfValues definition

lwm2m+json version

[{"bn":"/0/1/","n":"0","vs": <MQTT Server/Broker URI>},

{"n":"1","vb":false},

{"n":"2","v":0},

{"n":"3","vd":<PSK Identity>}, // opaque in Base64

{"n":"4","v": "<n/a>},

{"n":"5","vd": <Secret Key>}, // opaque in Base64

{"n":"10","v":1},

{"n":"26","vs": "/24/1"> },

{"n":"27","vs": "/23/1"> },

{"bn":"/1/1/","n":"0","v":1},

{"n":"1","v":86400},

{"n":"6","vb":false},

{"n":"7","vs":"M"},

{"bn":"/23/1/","n":"0","vs":<Key Identifier>},

{"n":"1","v":0},

{"n":"2","vs":<Secret Key>},

{"bn":"/24/1/","n":"6","vs":<Client Identifier>}

]

6.2. Registration Interface [100-199]

6.2.1. LightweightM2M-1.1-int-101 -- Initial Registration

Test Case Id

LightweightM2M-1.1-int-101

Test Object

Client and Server

Test Case Description

Test that the Client registers with the Server.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.1 as defined in Annex C.
- Device is turned on.
- The bootstrap procedure has been completed or the required bootstrap information is available to the Client.

Test Procedure

The Client automatically registers at the Server, once this information is available.

Normal flow:

1. Registration message (CoAP POST) is sent from Client to Server.

Pass-Criteria

1. In test step 1. , the Server receives the REGISTER command along with the following information:

- Endpoint Client Name (optional)
- registration lifetime
- LwM2M version (MUST be lwm2m=1.1)
- binding mode (optional)
- SMS number (optional)
- Objects and Object Instances (mandatory and optional objects / object instances) ; possibly with Version of Objects. The Security object (ID: 0) and the OSCORE object (ID: 21) MUST not be include in this list

2. In test step 1. , Client received "Success" message from Server (2.01 Created) related to the REGISTER command.

6.2.2. LightweightM2M-1.1-int-102 -- Registration Update

Test Case Id

LightweightM2M-1.1-int-102

Test Object

Client and Server

Test Case Description

Test that the client updates the registration information on the Server.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.1 as defined in Annex C
- Client is registered with the Server
- The Server will be prepared to change the Client lifetime registration to 20 sec to set the conditions for a Client UPDATE operation.
- On option, the Client will be prepared to update its registration before the registration time expires.

Test Procedure

The Client is registered with the Server and updates its registration according to the change of its registration information (Lifetime) (UPDATE Operation)

Normal flow:

1. The Server set the lifetime resource of the Server Object Instance to 20 sec (CoAP PUT /1/0/1)
2. UPDATE (Registration) message (CoAP POST) is sent from Client to Server with Lifetime parameter set to 20 sec
3. On option, before the registration expires (20 sec) the Client send a new UPDATE message without parameter to the Server

Pass-Criteria

1. In test step 1., the Server received a Success Message (2.04 Changed) related to its setting request
2. In test step 2., Server has received UPDATE operation with lifetime parameter = 20 sec
3. In test step 2., Client has received "Success" (2.04) message from Server related to the UPDATE command
4. In test step 3., either the Server received an UPDATE operation with no parameter or a de_registratoin occurs in the Server after the 20s

6.2.3. LightweightM2M-1.1-int-103 -- Deregistration

Test Case Id

LightweightM2M-1.1-int-103

Test Object

Client and Server

Test Case Description

Test that the client is able to deregister at the Server.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.3 as defined in Annex C
- Client is registered with the Server

Test Procedure

Client will no longer be available, thus, it SHOULD de-register

Normal flow:

1. The Server sends EXECUTE command on the Disable Resource of the Server Object (ID:1) Instance.
2. Deregistration message (CoAP DELETE) is sent from Client to Server.

Pass-Criteria

1. In test step 1., the Server receives the "Success" message (2.04 Changed) from the Client
2. In test step 2., the Client receives "Success" message (2.02 Deleted) from the Server
3. Client is removed from the servers registration database

6.2.4. LightweightM2M-1.1-int-104 -- Registration Update Trigger

Test Case Id

LightweightM2M-1.1-int-104

Test Object

Client and Server

Test Case Description

Test that the Client updates its registration with the Server when triggered with the Registration Update Trigger (see LwM2M Server object)

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.1 as defined in Annex C
- Device is turned on
- The bootstrap procedure has been completed or the required bootstrap information is available to the client
- The Client is registered with the Server with a Lifetime of 20 sec (LightweightM2M-1.1-int-102)

Test Procedure

Before Client Registration lifetime expired (lifetime is 20 sec) the Server sends Registration Update Trigger and the Client updates its registration via the UPDATE Operation

Normal flow:

1. Before Client registration expires on the Server (for test purposes a short registration lifetime is chosen 20 sec) a Registration Update Trigger message CoAP POST /1/0/8 is sent from Server to Client
2. UPDATE (Registration) message (CoAP POST) is sent from Client to Server

Pass-Criteria

1. In test step 1., Client received a Registration Update Trigger

2. In test step 1., Server received a "Success" message (2.04 Changed) related to the EXECUTE command (Registration Update Trigger).
3. In test step 2., Server received UPDATE operation without parameter
4. In test step 2., Client has received "Success" message (2.04 Changed) from Server related to its UPDATE message
5. After test step 2., the Client is still registered in the Server while the initial Registration lifetime expired

6.2.5. LightweightM2M-1.1-int-105 - Discarded Register Update

Test Case Id

LightweightM2M-1.1-int-105

Test Object

Client and Server

Test Case Description

Test if Server replies correctly (with error message) upon receiving an unexpected Register Update (e.g. after Lifetime expires). Test if Client respects the error message and fallbacks full registration (re-register).

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.1 as defined in Annex C
- The bootstrap procedure has been completed or the required bootstrap information is available to the client
- The Client is registered to LwM2M Server

Test Procedure

1. From LwM2M Server set a Client lifetime to some low value (e.g. 60 seconds), by sending a Write (CoAP POST) on /1/x/1 (where x is LwM2M Server Object Instance)
2. Make Server consider Device as deregistered (e.g. set expected lifetime to 1 second, or delete device/location on Server, or if in no-sec mode - spoof de-register message)
3. Wait for Client to perform Register Update (CoAP POST on /rd/{location})

Pass-Criteria

1. In test step 1, the Client accepts the lifetime change and respects it by sending a Register Update within lifetime
2. In test step 2, Server considers Device as de-registered. Client is not aware of de-registration at this point.

3. In test step 3, Register Update is discarded with CoAP Error 4.04 Not Found response.
4. Upon Receiving the above error message - Client performed full registration (CoAP POST on /rd, as described in *LightweightM2M-1.1-int-101*).

6.2.6. LightweightM2M-1.1-int-106 -- TCP Binding

Test Case Id

LightweightM2M-1.1-int-106

Test Object

Client and Server

Test Case Description

Test that the Client registers with the Server over TCP Binding.

Tool

n/a

Test code

n/a

Preconditions

- Device is turned on.
- The bootstrap procedure has been completed or the required bootstrap information (LwM2M Server over TCP binding and according LwM2M Security information) is available to the Client.

Test Procedure

1. Make the Client register to the Server

Pass-Criteria

1.) the Client established a TCP session with LwM2M Server.

b) Once Session is established - the CSM (Capabilities and Settings Messages) are exchanged between two endpoints (Client and Server)

c) the Client then sends a REGISTER message (CoAP POST on /rd with appropriate query) towards the Server along with the following information:

- Endpoint Client Name (optional)
- registration lifetime
- LwM2M version
- binding mode = T

- SMS number (optional)
- Objects and Object Instances (mandatory and optional objects / object instances) ; possibly with Version of Objects. Payload type of Object list SHOULD be CoRE Link Format (application/link-format), for example: </1/2>,</2>, </3/0>,</4/0>,</5/0>,</6/0>,</7/0>,</10>;ver="1.1"

Note: Object /0 and /21 MUST not be part of this list

d) the Client received "Success" message from Server (CoAP 2.01 Created) related to the REGISTER operation.

6.2.7. LightweightM2M-1.1-int-107 -- Extending the lifetime of a registration

Test Case Id

LightweightM2M-1.1-int-107

Test Object

Client and Server

Test Case Description

Test that Server can extend the LwM2M lifetime of Client.

Tool

n/a

Test code

n/a

Preconditions

- Device is turned on.
- The bootstrap procedure has been completed or the required bootstrap information (LwM2M Server configured with U mode) is available to the Client.

Test Procedure

1. Make the Client register to the Server with default U mode (CoAP POST on /rd with appropriate query) with lifetime of 60 seconds
2. Server changes the lifetime of the client to some long value (e.g. 120), by sending a Write (CoAP PUT) on /1/x/1 (where x is LwM2M Server Object Instance) with value of 120
3. Wait for next Register Update from Device

Pass-Criteria

1. Client has successfully registered to the Server, Registration is confirmed by Server with 2.01 Created message
2.) Client accepted a command by sending a *CoAP 2.04 Changed* response.

b) Client immediately Updated its Registration with the payload indicating that Lifetime has changed, i.e. CoAP POST on

/rd/{location}?lt=120

c) Server parsed and accepted the change of Lifetime and replied with *CoAP 2.04 Changed*.

3. Register Update arrives to the Server within lifetime

6.2.8. LightweightM2M-1.1-int-108 -- Turn on Queue Mode

Test Case Id

LightweightM2M-1.1-int-108

Test Object

Client and Server

Test Case Description

Test that Device can start working in Queue mode

Tool

n/a

Test code

n/a

Preconditions

- Device is turned on.
- The bootstrap procedure has been completed or the required bootstrap information (LwM2M Server configured with U mode) is available to the Client.

Test Procedure

1. Make the Client register to the Server with U binding and Queue mode activated (CoAP POST on /rd with ?b=U&Q query) The rest of the parameters is as described in *LightweightM2M-1.1-int-101*

Pass-Criteria

1. Client has successfully registered to the Server, Registration is confirmed by Server with 2.01 Created message. Server considers the Client to be in Queue Mode.

6.2.9. LightweightM2M-1.1-int-109 -- Behavior in Queue Mode

Test Case Id

LightweightM2M-1.1-int-109

Test Object

Client and Server

Test Case Description

Test that Server respects the Queue Mode and considers Client reachable only after communication initiated by Client

Tool

n/a

Test code

n/a

Preconditions

- Device is turned on.
- The bootstrap procedure has been completed or the required bootstrap information (LwM2M Server configured with U mode) is available to the Client.
    - MAX_TRANSMIT_WAIT on both Client and Server is configured to default value of 93 seconds.

Test Procedure

1. Make the Client register to the Server with U binding and Queue mode activated (CoAP POST on /rd with ?b=U&Q query) The rest of the parameters is as described in LightweightM2M-1.1-int-101
2. Server changes the lifetime of the client to some long value (e.g. 240), by sending a Write (CoAP POST) on /1/x/1 (where x is LwM2M Server Object Instance) with value of 240
3. Wait for 120 seconds (to make MAX_TRANSMIT_WAIT timer expire)
4. On a Server order an exemplary Read (for example on Lifetime resource /1/x/1)
5. Wait for the next Register Update from Device

Pass-Criteria

1. Client has successfully registered to the Server, Registration is confirmed by Server with 2.01 Created message
2.) Client accepted a command by sending a *CoAP 2.04 Changed* response.

b) Client immediately Updated its Registration with the payload indicating that Lifetime has changed, i.e. CoAP POST on/rd/{location}?lt=240

c) Server parsed and accepted the change of Lifetime and replied with *CoAP 2.04 Changed*.

3. After MAX_TRANSMIT_WAIT expires - client is allowed to go to sleep and SHOULD be considered not reachable by the Server
4. o message is sent to the Client at this point. Server queued the message.
5. Register Update is accepted by the Server. Client initiated communication triggers the Server to send all of the queued messages. Server sends the Read (CoAP GET) on /1/x/1

6.2.10. LightweightM2M-1.2-int-110 -- Initial Registration using Profile ID

Test Case Id

LightweightM2M-1.2-int-110

Test Object

Client and Server

Test Case Description

Test that the Client registers with the Server using profile ID

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.1 as defined in Annex C.
- Client supports use of 32 bit Profile ID in registration using default hash algorithm of SHA256.
- Server supports profile ID usage and has Configuration C.1 on the list of possible configurations that clients can register with profile ID, i.e. can match 32 bit truncated SHA256 hash value for configuration C.1 to what it receives from a client
- Device is turned on.
- The bootstrap procedure has been completed or the required bootstrap information is available to the Client.

Test Procedure

The Client automatically registers at the Server, once this information is available.

Normal flow:

1. Registration message (CoAP POST) is sent from Client to Server.

Pass-Criteria

1. In test step 1. , the Server receives the REGISTER command along with the following information:

- Endpoint Client Name (optional)
- registration lifetime
- LwM2M version (MUST be lwm2m=1.2)
- binding mode (optional)
- Profile ID (pid=6:27354b9a for object lists /1/0 and 3/0)
- Objects and Object Instances not present

2. In test step 1. , Client received "Success" message from Server (2.01 Created) related to the REGISTER command.

6.2.11. LightweightM2M-1.2-int-111 -- Initial Registration Using Mix of Profile ID and Object List

Test Case Id

LightweightM2M-1.2-int-111

Test Object

Client and Server

Test Case Description

Test that the Client registers with the Server using profile ID

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.5 as defined in Annex C.
- Client supports use of 32 bit Profile ID in registration using default hash algorithm of SHA256.
- Server supports profile ID usage and has Configuration C.1 on the list of possible configurations that clients can register with profile ID, i.e. can match 32 bit truncated SHA256 hash value for configuration C.1 to what it receives from a client
- Device is turned on.
- The bootstrap procedure has been completed or the required bootstrap information is available to the Client.

Test Procedure

The Client automatically registers at the Server, once this information is available.

Normal flow:

1. Registration message (CoAP POST) is sent from Client to Server.

Pass-Criteria

1. In test step 1. , the Server receives the REGISTER command along with the following information:

- Endpoint Client Name (optional)
- registration lifetime
- LwM2M version (MUST be lwm2m=1.2)

- binding mode (optional)
- Profile ID (pid=6:27354b9a for object lists /1/0 and 3/0)
- List of objects and object instances that are not part of Profile ID; <lwm2m=1.2;lt=86400;pid=6:27354b9a,/4/0>

2. In test step 1. , Client received "Success" message from Server (2.01 Created) related to the REGISTER command.

6.3. Device management & Service Enablement Interface [200-299]

6.3.1. LightweightM2M-1.1-int-201 -- Querying basic information in Plain Text format

Test Case Id

LightweightM2M-1.1-int-201

Test Object

Client and Server

Test Case Description

Querying the following data on the client (Device Object: ID 3) using Plain Text data format

- Manufacturer Name
- Model number
- Serial number Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client is supporting Plain Text data format (0)

Test Procedure

READ operations from Server on the targeted Resources have been received by the client.

Normal flow:

1. Successive READ (CoAP GET) operations are performed on the targeted Resources of the Device Object (ID:3) Instance, with the CoAP Accept option set to Plain Text data format (0)

Pass-Criteria

1. In test step 1, Server has received all the expected "Success" messages (2.05 Content) along with the requested

information with the expected values and according to the Plain Text data format:

- Manufacturer Name (ID:0)
- Model number (ID:1)
- Serial number (ID:2)

6.3.2. LightweightM2M-1.1-int-202 -- Querying basic information in Opaque format

paragraph

6.3.3. LightweightM2M-1.1-int-203 -- Querying basic information in TLV format

Test Case Id

LightweightM2M-1.1-int-203

Test Object

Client and Server

Test Case Description

Querying the Resources Values of Device Object ID:3 on the Client Resources Values of Device Object ID:3

Tool

n/a

Test code

n/a

Preconditions

- The client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server

Test Procedure

A READ operation from Server on the Device Object Instance has been received by the Client.

Normal flow:

1. READ (CoAP GET) operation on Device Object Instance (/3/0), with the CoAP Accept option set to TLV data format (11542)
2. Bits 7-6=00= Object Instance in which case the Value contains one or more Resource TLVs
3. Bits 7-6=11= Resource with Value

Pass-Criteria

1. In test step 1. Server has received the "Success" message (2.05 Content), along with the requested information in

the TLV data format (11542) and containing the expected values concerning:

- Manufacturer Name (ID:0)
- Model number (ID:1)
- Serial number (ID:2)
- Firmware Version (ID:3)
- Error Code (ID:11)
- Supported Binding and Modes (ID:16) ("U")

6.3.4. LightweightM2M-1.1-int-204 -- Querying basic information in JSON format

Test Case Id

LightweightM2M-1.1-int-204

Test Object

Client and Server

Test Case Description

Querying the Resources Values of Device Object ID:3 on the Client using JSON data format

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is supporting JSON data format (11543)
- Client is registered with the LwM2M Server

Test Procedure

A READ operation from Server on the Device Object Instance has been received by the Client.

Normal flow:

1. READ (CoAP GET) operation on the Device Object Instance (/3/0), with the CoAP Accept option set to JSON data format (11543)

Pass-Criteria

1. In test step 1. Server has received the success message (2.05 Content) along with the requested information in JSON data format (11543) and containing the expected values concerning:

- Manufacturer Name (ID:0)
- Model number (ID:1)
- Serial number (ID:2)
- Firmware Version (ID:3)
- Error Code (ID:11)
- Supported Binding and Modes (ID:16) ("U")

6.3.5. LightweightM2M-1.1-int-205 -- Setting basic information in Plain Text format

Test Case Id

LightweightM2M-1.1-int-205

Test Object

Client and Server

Test Case Description

Setting writable Resources of Server Object (ID:1) Instance 0 using Plain Text data format (0)

Preconditions

- The Client supports the Configuration C.3 as defined in Annex C
- Client is supporting Plain Text data format (0)
- Client is registered with the Server

Test Procedure

Successive WRITE operations from Server on the Resources of the Server Object (ID:1) Instance have been received by the client. This test has to set the following resources with specific values:

- Default minimum period (ID:2) : 0101 sec
- Default maximum period (ID:3) : 1010 sec
- Disable timeout (ID:5) : 2000 sec

Normal flow:

1. Successive WRITE (CoAP PUT) operations are performed on the Resources of the Instance 0 of the Server Object in using the predefined values above. The Plain Text data format (0) is used
2. The Server verifies (READs/CoAP GET) the result of the WRITE operations by querying the Instance 0 of the Server Object in using the TLV data format.
3. The steps 1., 2. of the test are re-played with the initial values of the Configuration 3

Pass-Criteria

1. In test step 1. the Server receives a "Success" message (2.04 Changed) for each WRITE operation
2. In test step 2, Server received the "Success" message (2.05 Content), along with the requested information in the TLV data format and containing the expected values defined for this test
3. In test step 3, the final stage confirms that the initial Configuration C.3 has been restored in the Client.

6.3.6. LightweightM2M-1.1-int-210 -- Setting basic information in Opaque format

6.3.7. LightweightM2M-1.1-int-211 -- Querying basic information in CBOR format

Test Case Id

LightweightM2M-1.1-int-211

Test Object

Client and Server

Test Case Description

Querying the following data on the client (Server Object: ID 1) using CBOR format

- Short Server ID (ID: 0)
- Notification Storing When Disabled or Offline (ID: 6)
- Binding (ID: 7)

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server in LwM2M 1.1
- Client is supporting CBOR data format (60)

Test Procedure

Successive READ operations from Server on the targeted Resources have been received by the client.

Normal flow:

1. Successive READ (CoAP GET) operations are performed on the targeted Resources of the Server Object (ID: 1) Instance, with the CoAP Accept option set to CBOR data format (60)

Pass-Criteria

1. In test step 1. Server has received the expected "Success" messages (2.05 Content) along with the requested information with the expected values and according to the CBOR data format:

- Short Server ID (ID: 0): 1
- Notification Storing When Disabled or Offline (ID: 6): false
- Binding (ID: 7): "U"

6.3.8. LightweightM2M-1.1-int-212 -- Setting basic information in CBOR format

Test Case Id

LightweightM2M-1.1-int-212

Test Object

Client and Server

Test Case Description

Setting writable Resources of Server Object (ID:1) Instance 0 using CBOR data format (60)

Preconditions

- The Client supports the Configuration C.3 as defined in Annex C
- Client is supporting CBOR data format (60)
- Client is registered with the Server in LwM2M 1.1

Test Procedure

Successive WRITE operations from Server on the Resources of the Server Object (ID:1) Instance have been received by the client. This test has to set the following resources with specific values:

- Default minimum period (ID:2): 0101 sec
- Default maximum period (ID:3): 1010 sec
- Notification Storing When Disabled or Offline (ID: 6) : true

Normal flow:

1. Successive WRITE (CoAP PUT) operations are performed on the Resources of the Instance 0 of the Server Object in using the predefined values above. The CBOR data format (60) is used
2. The Server verifies (READs/CoAP GET) the result of the WRITE operations by querying the Instance 0 of the Server Object in using client preferred data format.

Pass-Criteria

1. In test step 1. the Server receives a "Success" message (2.04 Changed) for each WRITE operation
2. In test step 2, Server received the "Success" message (2.05 Content), along with the requested information in client preferred data format and containing the expected values defined for this test

6.3.9. LightweightM2M-1.1-int-215 -- Setting basic information in TLV format

Test Case Id

LightweightM2M-1.1-int-215

Test Object

Client and Server

Test Case Description

Setting writable Resources of Server Object (ID:1) Instance 0 in using the TLV data format (11542) and restoring these Resources to their initial values

Preconditions

- The Client supports the Configuration C.3 as defined in Annex C
- The Client is registered with the Server
- The current values of the Server Object (ID:1) Instance 0, are saved on the Server

Test Procedure

A WRITE operation from Server on the Server Object (ID:1) Instance 0 has been received by the client using TLV data format (11542). The set of values to write is given in the sample below (215-SetOfValues)

Normal flow:

1. A 1st WRITE (CoAP POST) operation on the Server Object (ID:1) Instance 0 is performed in using the set of values contains in the 215-SetOfValues sample below.The data format TLV is used. (11542)
2. The Server READs the result of the WRITE operation by querying the Server Object (ID:1) Instance 0.
3. A 2nd WRITE (CoAP PUT) operation on the Server Object (ID:1) Instance 0 is performed in using the Initial values preserved on the Server in conformance to the test pre-conditions.
4. The Server READs the result of the previous WRITE operation by querying the Server Object (ID:1) Instance 0.

Pass-Criteria

1. The Server receives the success messages for the steps 1.(2.04), 2. (2.05), 3. (2.04) and 4. (2.05) of the test
2. In test step 4. , the received values are consistent with the pre-defined values of the 215-SetOfValues sample
3. In test step 6, the received values are consistent with the values present in the initial Configuration C.3

215-SetOfValues definition

lwm2m+tlv version

C1 02 65 ; /1/0/2 101sec

C2 03 03 F2 ; /1/0/3 1010 sec

C2 05 07 D0 ; /1/0/5 2000 sec

C1 06 01 ; /1/0/6 true

C1 07 55 ; /1/0/7 U

6.3.10. LightweightM2M-1.1-int-220 -- Setting basic information in JSON format

Test Case Id

LightweightM2M-1.1-int-220

Test Object

Client and Server

Test Case Description

Setting writable Resources of Server Object (ID:1) Instance 0 in using the JSON data format (11543) and restoring these Resources to their initial values

Preconditions

- The Client supports the Configuration C.3 as defined in Annex C
- The Client is registered with the Server
- The current values of the Server Object (ID:1) Instance 0, are saved on the Server

Test Procedure

A WRITE operation from Server on the Server Object (ID:1) Instance 0 has been received by the client using JSON data format (11543). The set of values to write is given in the sample below (220-SetOfValues)

Normal flow:

1. A 1^st^ WRITE (CoAP POST) operation on the Server Object (ID:1) Instance 0 is performed in using the set of values contains in the 220-SetOfValues sample below. The data format JSON is used. (11543)
2. The Server READs the result of the WRITE operation by querying the the previously targeted Resources of the Server Object (ID:1) Instance 0.
3. A 2^nd^ WRITE (CoAP PUT) operation on the Server Object (ID:1) Instance 0 is performed in using the Initial values which have been preserved (pre-conditions).
4. The Server READs the result of the previous WRITE operation by querying the Server Object (ID:1) Instance 0.

Pass-Criteria

1. The Server receives the correct status codes for the steps 1. (2.04), 2. (2.05), 3. (2.04) and 4. (2.05) of the test.
2. In test step 2. , the received values are consistent with the 220-SetOfValues sample
3. In test step 4, the received values are consistent with the values present in the initial Configuration 3

220-SetOfValues definition

lwm2m+json version

[{"bn":"/",**"n":"1/0/2","v"**:0101},

** {"n":"1/0/3","v":1010},**

** {"n":"1/0/5","v":2000},**

** {"n":"1/0/6","bv":true},**

** {"n":"1/0/7","sv":"U"}**]

6.3.11. LightweightM2M-1.1-int-221 -- Attempt to perform operations on Security Object (ID: 0)

Test Case Id

LightweightM2M-1.1-int-221

Test Object

Client and Server

Test Case Description

Test, that Client rejects any attempt of reading or writing Security Object (ID: 0)

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server

Test Procedure

1. Server performs a Read (*CoAP GET*) on LwM2M Security Object (/0).
2. Server performs a Write (*CoAP PUT*) on LwM2M Security Object (/0) Resource LwM2M Server URI (/0/0/0)
3. Server performs a Write-Attributes(*CoAP PUT*) on LwM2M Security Object (/0):

a. pmin=30,

b. pmax=45.

Pass-Criteria

1. Client did not allow to read on LwM2M Security Object and replied with *CoAP 4.01 Unauthorized* error
2. Client did not allow to Write on LwM2M Security Object and replied with *CoAP 4.01 Unauthorized* error
3. Client did not allow to Write-Attributes on LwM2M Security Object and replied with *CoAP 4.01 Unauthorized* error

6.3.12. LightweightM2M-1.1-int-222 -- Read on Object

Test Case Id

LightweightM2M-1.1-int-222

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with the LwM2M Read operation on whole Object

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server

Test Procedure

1. Server performs a Read (*CoAP GET*) on LwM2M Server Object (/1).
2. Server performs a Read (*CoAP GET*) on Device Object (/3).

Pass-Criteria

1. Client replied with CoAP 2.05 Content. Message payload contains the Object Instance and all of the Object Resources available on Client.
2. Client replied with CoAP 2.05 Content. Message payload contains only the single Object Instance and all of the Object Resources available on Client.

6.3.13. LightweightM2M-1.1-int-223 -- Read on Object Instance

Test Case Id

LightweightM2M-1.1-int-223

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with the LwM2M Read operation on Object Instance

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server

Test Procedure

1. Server performs a Read (*CoAP GET*) on LwM2M Server Object (/1/x, where x in a LwM2M Server Object Instance).
2. Server performs a Read (*CoAP GET*) on Device Object Instance 0 (/3/0).

Pass-Criteria

1. Client replied with CoAP 2.05 Content. Message payload contains the Object Instance and all of the Object Resources available on Client.
2. Client replied with CoAP 2.05 Content. Message payload contains the Object Instance and all of the Object Resources available on Client.

6.3.14. LightweightM2M-1.1-int-224 -- Read on Resource

Test Case Id

LightweightM2M-1.1-int-224

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with the LwM2M Read operation on Resource level

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports *LwM2M Server* Object mandatory Resources -- Short Server ID, Lifetime, Notification Storing and

Binding)

Test Procedure

1. Server performs separate Reads (*CoAP GET*) on LwM2M Server Object mandatory Resources:

- Short Server ID (1/x/0, where x in a LwM2M Server Object Instance),
- Lifetime (1/x/1),
- Notification Storing (1/x/6),
- Binding (1/x/7),

2. Server performs a Read (*CoAP GET*) on Device Object mandatory Resource:

- Supported Binding and Modes (/3/0/16)

Pass-Criteria

1. Client replied with CoAP 2.05 Content for each of Read requests. Message payload contains the Resource values.
2. Client replied with CoAP 2.05 Content. Message payload contains the Resource value.

6.3.15. LightweightM2M-1.1-int-225 -- Read on Resource Instance

Test Case Id

LightweightM2M-1.1-int-225

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with the LwM2M Read operation on Resource Instance level

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server

Test Procedure

1. Server performs Reads (*CoAP GET*) on Instance of mandatory Resource in Device Object - **Error Code** (/3/0/11/x, where x is a Resource Instance present on Client).

Pass-Criteria

1. Client replied with CoAP 2.05 Content for Read requests. Message payload contains the Resource Instance /3/0/11/x value only.

6.3.16. LightweightM2M-1.1-int-226 -- Write (Partial Update) on Object Instance

Test Case Id

LightweightM2M-1.1-int-226

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with the LwM2M Write operation on Object Instance

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports mandatory resources of LwM2M Server Object

Test Procedure

1. Server performs Write (*CoAP POST - Partial Update*) on LwM2M Server Object (/1/x, where x in a LwM2M Server Object Instance) with the following values:
- Lifetime: 61 seconds
- Notification storing: *true*
2. Server performs a Read (*CoAP GET*) on LwM2M Server Object and check on Clients response
3. Server performs a Write (*CoAP POST*) on LwM2M Server Object (/1/x, where x in a LwM2M Server Object Instance) to restore previous values (lifetime, notification storing)

Pass-Criteria

1. Server Write is sent in single message pointing on Object Instance. Client replied with CoAP 2.04 Changed , accepted the operation and send the Registration Update (CoAP POST) with updated lifetime.
2. Client replied with CoAP 2.05 Content. Message payload contains the Resource Values set in previous point

3. Client replied with CoAP 2.04 Changed and accepted the data

6.3.17. LightweightM2M-1.1-int-227 -- Write (replace) on Resource

Test Case Id

LightweightM2M-1.1-int-227

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with the LwM2M Write operation on Resource level

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports mandatory resources of LwM2M Server Object

Test Procedure

1. Server performs a Write (*CoAP PUT*) on LwM2M Server Object Lifetime Resources (/1/x/1, where x in a LwM2M Server Object Instance) with the following values:
- Lifetime: 63 seconds
2. Server performs a Read (*CoAP GET*) on LwM2M Server Object's Lifetime Resource
3. Server performs a Write (*CoAP PUT*) on LwM2M Server Object Lifetime Resources (/1/x, where x in a LwM2M Server Object Instance) to restore previous Lifetime value

Pass-Criteria

1. Client replied with CoAP 2.04 Changed, accepted the operation and send the Registration Update (CoAP POST) with updated lifetime.
2. Client replied with CoAP 2.05 Content. Message payload contains the Lifetime value set in previous point
3. Client replied with CoAP 2.04 Changed, accepted the operation and send the Registration Update (CoAP POST) with updated lifetime.

6.3.18. LightweightM2M-1.1-int-228 -- Write on Resource Instance

Test Case Id

LightweightM2M-1.1-int-228

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with the LwM2M Write operation on Resource Instance level

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports non-mandatory Portfolio (ID: 16) Object
- Portfolio Object has an Instance with Identity (ID: 0) Resource with at least two Resource Instances

Test Procedure

1. Server performs a Write (*CoAP POST*) on Portfolio Object's Identity Resource instance (/16/x/0/y, where x in a Portfolio Object Instance and y is an Identity Resource Instance) with "test" value.
2. Server performs a Read (*CoAP GET*) on Portfolio Object's Identity Resource Instance (/16/x/0/y, where x in a Portfolio Object Instance and y is an Identity Resource Instance)

Pass-Criteria

1. Server message contains the data only for "y" Resource Instance, the other instance is not mentioned in payload. Client replied with CoAP 2.04 Changed and accepted the new data
2. Client replied with CoAP 2.05 Content. Message payload contains the data only for "y" Resource Instance, the other instances is not mentioned in payload. Resource Instance value matches the one set in previous point.

6.3.19. LightweightM2M-1.1-int-229 -- Read-Composite Operation

Voided

See LightweightM2M-1.1-int-280 that specifies pass criteria in more detail

6.3.20. LightweightM2M-1.1-int-230 -- Write-Composite Operation

Note: this functionality is not simply available through usage of Core Objects. Client needs to support non-mandatory

Portfolio Object.

Test Case Id

LightweightM2M-1.1-int-230

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with Write-Composite operation\

Note: support of this operation is not mandatory

Tool

n/a

Test code

n/a

Preconditions

The Client supports the Configuration C.1 as defined in Annex C

Client is registered with the LwM2M Server

Client supports mandatory resources of *LwM2M Server* and *non-mandatory Portfolio (ID: 16) Object*

Test Procedure

1. Server performs a Write-Composite (*CoAP iPATCH*) on the following combination of Resources, Resources Instance:

- *LwM2M Server Object* (/1/x, where x in a LwM2M Server Object Instance) with the following values:
    - *Lifetime:* 61 seconds
    - *Notification storing:* true
- *Portfolio Object, Instance 0, resource Identity (ID: 0):*
    - *Identity.0 = aa (/16/0/0/0),*
    - *Identity.1 = bb (/16/0/0/1),*
    - *Identity.2 = cc (/16/0/0/2),*
    - *Identity.3 = dd (/16/0/0/3),*

2. Server performs Read (CoAP GET) on *LwM2M Server Object* and on *Portfolio* Object

Pass-Criteria

1. erver's request is sent in a single CoAP iPATCH message. Message payload contains requested data. Content format

is either: SenML CBOR or SenML JSON. Client accepted the request, replied with 2.04 Changed and send the Registration Update (CoAP POST) with updated lifetime.

2. Client replied correctly with 2.05 Content and the response payload matches the data written in step 1.

6.3.21. LightweightM2M-1.1-int-231 -- Querying basic information in SenML JSON format

Test Case Id

LightweightM2M-1.1-int-231

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with SenML JSON format for READ operations on O, OI, R, RI levels.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports mandatory resources of LwM2M Server (ID: 1) and Device (ID: 3) Object

Test Procedure

1. Server performs a Read operation (CoAP GET) on LwM2M Server (ID: 1) Object with CoAP Accept option set to SenML JSON (110).
2. Server performs a Read operation (CoAP GET) on Device Object Instance (/3/0) with CoAP Accept option set to SenML JSON (110).
3. Server performs a Read operation (CoAP GET) on Supported Binding and Modes resource of Device Object (/3/0/16) with CoAP Accept option set to SenML JSON (110).
4. Server performs a Read operation (CoAP GET) on Error Code resource instance of Device Object (/3/0/11/x, where x is a resource instance on Device) with CoAP Accept option set to SenML JSON (110).

Pass-Criteria

1. Client replied with CoAP 2.05 Content. Message payload contains the Object Instances and all of the Object Resources available on Client. Message Content-Format is SenML JSON (110)
2. Client replied with CoAP 2.05 Content. Message payload contains all of the Object Resources available on Client. Message Content-Format is SenML JSON (110)

3. Client replied with CoAP 2.05 Content. Message payload contains the Resources value. Message Content-format is SenML JSON (110)
4. Client replied with CoAP 2.05 Content. Message payload contains the Resources Instance value. Message Content-format is SenML JSON (110)

6.3.22. LightweightM2M-1.1-int-232 -- Querying basic information in SenML CBOR format

Test Case Id

LightweightM2M-1.1-int-232

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with SenML CBOR format for READ operations on O, OI, R, RI levels.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports mandatory resources of LwM2M Server (ID: 1) and Device (ID: 3) Object

Test Procedure

1. Server performs a Read operation (CoAP GET) on LwM2M Server (ID: 1) Object with CoAP Accept option set to SenML CBOR (112).
2. Server performs a Read operation (CoAP GET) on Device Object Instance (/3/0) with CoAP Accept option set to SenML CBOR (112).
3. Server performs a Read operation (CoAP GET) on Supported Binding and Modes resource of Device Object (/3/0/16) with CoAP Accept option set to SenML CBOR (112).
4. Server performs a Read operation (CoAP GET) on Error Code resource instance of Device Object (/3/0/11/x, where x is a resource instance on Device) with CoAP Accept option set to SenML CBOR(112).

Pass-Criteria

1. Client replied with CoAP 2.05 Content. Message payload contains the Object Instances and all of the Object Resources available on Client. Message Content-format is SenML CBOR(112).
2. Client replied with CoAP 2.05 Content. Message payload contains all of the Object Resources available on Client.

Message Content-format is SenML CBOR (112)

3. Client replied with CoAP 2.05 Content. Message payload contains the Resources value. Message Content-format is SenML CBOR (112)
4. Client replied with CoAP 2.05 Content. Message payload contains the Resources Instance value. Message Content-format is SenML CBOR (112)

6.3.23. LightweightM2M-1.1-int-233 -- Setting basic information in SenML CBOR format

Note: this functionality is not simply available through usage of Core Objects. Client needs to support non-mandatory Portfolio Object.

Test Case Id

LightweightM2M-1.1-int-233

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with SenML CBOR format for Write operations on OI, R and RI levels.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports LwM2M Server Object mandatory Resources -- Short Server ID, Lifetime, Notification Storing and Binding)
- Client supports non-mandatory Portfolio Object

Test Procedure

1. Server performs Write (CoAP POST) on LwM2M Server Object Instance (/1/x, where x in a LwM2M Server Object Instance) with the following values:

- Lifetime: 61 seconds
- Notification storing: true

Message Content-format SHOULD be set to SenML CBOR (112)

2. Server performs a Read (CoAP GET) on LwM2M Server Object

3. Server performs a Write (CoAP POST or CoAP PUT) on Instance 0 of Identity Resource in Portfolio Object (/16/0/0/0):

- Identity.0 = "test_value"

Message Content-format SHOULD be set to SenML CBOR (112)

The choice of Write Update (CoAP POST) or Write Replace (CoAP

PUT) is irrelevant for this test procedure since Write operation includes the mandataory resource instance.

4. Server performs a Read (CoAP GET) on Portfolio Object

5. Server performs a Write (CoAP POST or CoAP PUT) on LwM2M Server Object Lifetime Resource (/1/x, where x in a LwM2M Server Object Instance) with the following values:

- Lifetime: 63 seconds

Message Content-format SHOULD be set to SenML CBOR (112). The choice of Write Update (CoAP POST) or Write Replace (CoAP PUT) is irrelevant for this test procedure since Write operation includes the mandataory resource instance.

6. Server performs a Read (CoAP GET) on LwM2M Server Object's Lifetime resource

Pass-Criteria

1. Client replied with CoAP 2.04 Changed, accepted the operation and send the Registration Update (CoAP POST) with updated lifetime.
2. Client replied with CoAP 2.05 Content. Message payload contains the data which were written in step 1.
3. Client replied with CoAP 2.04 Changed and accepted the operation
4. Client replied with CoAP 2.05 Content. Message payload contains the data which were written in step 3.
5. Client replied with CoAP 2.04 Changed, accepted the operation and send the Registration Update (CoAP POST) with updated lifetime.
6. Client replied with CoAP 2.05 Content. Message payload contains the data which were written in step 5.

6.3.24. LightweightM2M-1.1-int-234 -- Setting basic information in SenML JSON format

Note: this functionality is not simply available through usage of Core Objects. Client needs to support non-mandatory Portfolio Object.

Test Case Id

LightweightM2M-1.1-int-234

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with SenML JSON format for Write operations on OI, R and RI levels.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports LwM2M Server Object mandatory Resources -- Short Server ID, Lifetime, Notification Storing and Binding)
- Client supports non-mandatory Portfolio Object

Test Procedure

1. Server performs Write (CoAP POST) on LwM2M Server Object Instance (/1/x, where x in a LwM2M Server Object Instance) with the following values:

- Lifetime: 61 seconds
- Notification storing: true

Message Content-format SHOULD be set to SenML JSON (110)

2. Server performs a Read (CoAP GET) on LwM2M Server Object
3. Server performs a Write (CoAP POST or CoAP PUT) on Instance 0 of Identity Resource in Portfolio Object (/16/0/0/0):

- Identity.0 = "test_value"

Message Content-format SHOULD be set to SenML JSON (110)

The choice of Write Update (CoAP POST) or Write Replace (CoAP PUT) is irrelevant for this test procedure since Write operation includes the mandataory resource instance.

4. Server performs a Read (CoAP GET) on Portfolio Object
5. Server performs a Write (CoAP POST or CoAP PUT) on LwM2M Server Object Lifetime Resource (/1/x, where x in a LwM2M Server Object Instance) with the following values:

- Lifetime: 63 seconds

Message Content-format SHOULD be set to SenML JSON (110)

The choice of Write Update (CoAP POST) or Write Replace (CoAP PUT) is irrelevant for this test procedure since Write operation includes the mandataory resource instance.

6. Server performs a Read (CoAP GET) on LwM2M Server Object's Lifetime resource

Pass-Criteria

1. Client replied with CoAP 2.04 Changed, accepted the operation and send the Registration Update (CoAP POST) with updated lifetime.
2. Client replied with CoAP 2.05 Content. Message payload contains the data which were written in step 1
3. Client replied with CoAP 2.04 Changed and accepted the operation
4. Client replied with CoAP 2.05 Content. Message payload contains the data which were written in step 3.
5. Client replied with CoAP 2.04 Changed, accepted the operation and send the Registration Update (CoAP POST) with updated lifetime.
6. Client replied with CoAP 2.05 Content. Message payload contains the data which were written in step 5.

6.3.25. LightweightM2M-1.1-int-235 -- Read-Composite Operation on root path

Test Case Id

LightweightM2M-1.1-int-235

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with Read-Composite operation on root path (whole Data Model)

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports mandatory resources of LwM2M Server and Device Objects

Test Procedure

1. Server performs a Read-Composite (CoAP FETCH) operation on / (root path) by including "/" in Read-Composite Payload Content.

"Content Format ID" is set to either SenML CBOR or SenML JSON

"Accept: Content-format" is set either to SenML CBOR or SenML JSON

Pass-Criteria

1. Client accepted the request and replied with 2.05 Content. Message payload contains all of the readable resources

available on Client. Response content format is either: SenML CBOR or SenML JSON, depending on what was requested by Server. Security Object (ID: 0), OSCORE Object (ID:21) and COSE object (ID:23) MUST not be part of the payload

6.3.26. LightweightM2M-1.1-int-236 -- Read-Composite - Partial Presence

Test Case Id

LightweightM2M-1.1-int-236

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with Read-Composite on list of Resources, where some of them does not exist on Client.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports mandatory resources of LwM2M Server and Device Objects

Test Procedure

1. Server performs a Read-Composite (CoAP FETCH) on both existing resources:

- LwM2M Server Object Instance (/1/x, where x in a LwM2M Server Object Instance),
- Lifetime Resource (/1/x/1),
- Error Code Resource Instance (/3/0/11/0).
- And non-existing resources, e.g.:
- Audio Clip.0.Clip (3339/0/5522)
- scellID.0.plmnID (3353/0/6030)

"Content Format ID" is set to either SenML CBOR or SenML JSON

"Accept: Content-format" is set either to SenML CBOR or SenML JSON

Pass-Criteria

1. Client accepted the request and replied with 2.05 Content. Message payload contains all of the readable resources available on Client. Response content format is either: SenML CBOR or SenML JSON, depending on what was requested by Server. The Read-Composite operation is treated as non-atomic and handled as best effort by the client. That is, if any of the requested resources do not have a valid value to return, they will not be included in the response

6.3.27. LightweightM2M-1.1-int-237 -- Read on Object without specifying Content-Type

Test Case Id

LightweightM2M-1.1-int-237

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with the LwM2M Read operation on whole Object, when Server does not specify the "Accept: Content Format" CoAP Option

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server

Test Procedure

1. Server performs a Read (CoAP GET) on LwM2M Server Object (/1) without specifying the "Accept: Content Format" CoAP Option
2. Server performs a Read (CoAP GET) on Device Object (/3) without specifying the expected Content-Type

Pass-Criteria

A. Client replied with CoAP 2.05 Content. Message payload contains the Object Instance and all of the Object Resources available on Client. Content-Type is of Client choice (either SenML CBOR or SenML JSON)

B. Client replied with CoAP 2.05 Content. Message payload contains only the single Object Instance and all of the Object Resources available on Client. Content-Type is of Client choice (either SenML CBOR or SenML JSON)

6.3.28. LightweightM2M-1.1-int-241 -- Executable Resource: Rebooting the device

Test Case Id

LightweightM2M-1.1-int-241

Test Object

Client and Server

Test Case Description

Test that the Device can be remotely reboot via the Device Object (ID:3)

Tool

n/a

Test code

n/a

Preconditions

- The client supports the Configuration C.1 as defined in Annex C
- Device is switched on
- Client is registered with the Server. (Client might be in a state that requires a reboot)

Test Procedure

An EXECUTE operation from Server on the resource Reboot of the Device Object (ID:3) has been received by the client.

Normal flow:

1. Server performs Execute (CoAP POST) operation on the Reboot Resource of Device Object (ID:3) Instance
2. Client registers again with the Server as in the test LightweightM2M-1.1-int-101

Pass-Criteria

1. In test step 1, the Server receives the success message (2.04 Changed) related to the EXECUTE operation on the Client
2. In test step2., Device reboots and the Client registers successfully with the Server again (see LightweightM2M-1.1-int-101)

6.3.29. LightweightM2M-1.1-int-256 -- Write Operation Failure

Test Case Id

LightweightM2M-1.1-int-256

Test Object

Client and Server

Test Case Description

Setting the Short Server ID Resource of Server Object (ID: 1) on the Client

Preconditions

The Client supports the Configuration C.1 as defined in Annex C

The Client is registered with the Server in LwM2M 1.1

Test Procedure

A Write (CoAP POST/PUT) operation is performed by the Server on the Short Server ID Resource of the Server Object (ID: 1).

This test has to set the following resources with specific values:

- Short Server ID (ID: 0): 123

Normal flow:

1. A WRITE is performed on Server Object (ID: /1) in using the predefined values above
2. A READ is performed on Short Server ID Resource of Server Object (ID: /1/0/0)

Pass-Criteria

1. In test step 1. Server has received the failure message (4.05 Method not allowed)
2. In test step 2. Server has received the success message (2.05 Content) along with the requested information and containing the expected values concerning:

- Short Server ID (ID: 0): 1

6.3.30. LightweightM2M-1.1-int-257 -- Write-Composite Operation

Test Case Id

LightweightM2M-1.1-int-257

Test Object

Client and Server

Test Case Description

Setting writable Resources of Server Object (ID:1) and Device object (ID: 3)

Preconditions

- The Client supports the Configuration C.6 as defined in Annex C
- Client is registered with the Server in LwM2M 1.1
- Client and Server MUST support Write composite operation

Test Procedure

A WRITE-COMPOSITE (CoAP iPATCH) operation from Server on the Server Object (ID:1) and Device object (ID: 3) has been received by the client. The set of values to write is given in the sample below (257-SetOfValues)

Normal flow:

1. The Server gets the initial configuration by querying (READs/CoAP GET) the Server Object and Device object values using its preferred data format
2. A WRITE-COMPOSITE (CoAP iPATCH) operation is performed on the Instance 0 of the Server Object in using the set of values to write is given in the sample below (257-SetOfValues)
3. The Server verifies the result of the WRITE operations by querying (READs/CoAP GET) the Server Object and Device object values using its preferred data format

Pass-Criteria

1. In test step 1. the Server received the "Success" message (2.05 Content), with the values present in the initial Configuration 6.
2. In test step 2. the Server receives a "Success" message (2.04 Changed) for WRITE-COMPOSITE operation
3. In test step 3. Server received the "Success" message (2.05 Content), with the values present in the 257-SetOfValues sample

257-SetOfValues definition

lwm2m+senml+json version

[{"bn":"/","n":"1/0/2","v":101},

{"n":"1/0/6","vb":true},

{"n":"3/0/13","v":0}]

lwm2m+senml+cbor version

83 # array(3)

A3 # map(3)

21 # negative(1)

61 # text(1)

2F # "/"

00 # unsigned(0)

65 # text(5)

312F302F32 # "1/0/2"

02 # unsigned(2)

18 65 # unsigned(101)

A2 # map(2)

00 # unsigned(0)

65 # text(5)

312F302F36 # "1/0/6"

04 # unsigned(4)

F5 # primitive(21)

A2 # map(2)

00 # unsigned(0)

66 # text(6)

332F302F3133 # "3/0/13"

02 # unsigned(2)

00 # unsigned(0)

6.3.31. LightweightM2M-1.1-int-260 -- Discover Command

Test Case Id

LightweightM2M-1.1-int-260

Test Object

Client and Server

Test Case Description

Discovering the implemented Resources of the Device Object as well as reporting the various Attributes attached to Object / Object Instance and Resources

Preconditions

- The Client supports the Configuration C.4 as defined in Annex C
- The Client is registered with the Server
- No <NOTIFICATION> Attribute is initially attached to the Object Device

Test Procedure

Various DISCOVER operations are performed by the Server on the Object Device. Several WRITE-ATTRIBUTES commands are interleaved to modify the content of the DISCOVER command replies

Normal flow:

1. The initial DISCOVER (CoAP GET Accept:40) operation is performed on Objet Device ID:3
2. A WRITE-ATTRIBUTES operation set the pmin=10 & pmax=200 <NOTIFICATION> Attributes at the Object level
3. DISCOVER command is performed at Object Instance level of Object ID:3
4. The WRITE-ATTRIBUTES operation set the lt=1, gt=6 et st=1 <NOTIFICATION> Attributes of the Resource ID:7 (Power Source Voltage)

5. Same DISCOVER command as in test step 3. is performed
6. DISCOVER operation is performed on Resource ID:7 of the Object Device Instance

Pass-Criteria

1. In test step 1, the Success Message ("2.05" Content) is received by the Server along with the payload related to the DISCOVER request and containing :

< /3>,</3/0>,</3/0/1>, </3/0/2>, </3/0/3>, </3/0/4>,</3/0/6>;dim=2,</3/0/7>;dim=2,</3/0/8>;dim=2,</3/0/9>, </3/0/11>,</3/0/16>

2. In test step 2, a Success message is received by the Server ("2.04" Changed) related to the WRITE-ATTRIBUTES operation
3. In test step 3, the Success message ("2.05" Content) is received by the Server along with the payload related to the DISCOVER request and containing :

</3/0>,</3/0/1>, </3/0/2>, </3/0/3>, </3/0/4>, </3/0/6>;dim=2, </3/0/7>;dim=2, </3/0/8>;dim=2, </3/0/9>,</3/0/11>, </3/0/16>

4. In test step 4, a "Success" message is received by the Server ("2.04" Changed) related to the WRITE-ATTRIBUTES operation
5. In test step 5, the Success message ("2.05" Content) is received by the Server along with the same payload received in step 3, except for the Resource 7 :

</3/0>,</3/0/1>, </3/0/2>, </3/0/3>, </3/0/4>, </3/0/6>;dim=2, </3/0/7>;dim=2, ;lt=1;gt=6;st=1, </3/0/8>;dim=2, </3/0/9>, </3/0/11>,</3/0/16>

6. In test step 6, the Success message ("2.05" Content) is received by the Server along with the payload related to the DISCOVER request and just containing :

</3/0/7>;pmin=10;pmax=200;dim=2 ;lt=1;gt=6 ;st=1

6.3.32. LightweightM2M-1.1-int-261 -- Write-Attribute Operation on a multiple resource

Test Case Id

LightweightM2M-1.1-int-261

Test Object

Client and Server

Test Case Description

Discovering the implemented Resources of the Device Object as well as reporting the various Attributes attached to Object / Object Instance and Resources

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- The Client is registered with the Server in LwM2M 1.1

- No <NOTIFICATION> Attribute is initially attached to the Object Device

Test Procedure

Various DISCOVER operations are performed by the Server on the Object Device. Several WRITE-ATTRIBUTES commands are interleaved to modify the content of the DISCOVER command replies

Normal flow:

1. A DISCOVER operation is performed on Error Code resource (ID: 11) of the Object Device (ID: 3)
2. A WRITE-ATTRIBUTES operation set the pmin=10&pmax=200 <NOTIFICATION> Attributes at the Object level (ID: /3)
3. A WRITE-ATTRIBUTES operation set the pmax=320 <NOTIFICATION> Attributes at the Object Instance level (ID: /3/0)
4. A WRITE-ATTRIBUTES operation set the pmax=100&epmin=1&epmax=20 <NOTIFICATION> Attributes at the Resource Instance level (ID: /3/0/11/0)
5. The same DISCOVER than in step 1. is performed

Pass-Criteria

1. In test step 1, the Success Message ("2.05" Content) is received by the Server along with the payload related to the DISCOVER request and containing:

</3/0/11>;dim=1,</3/0/11/0>

2. In test step 2., 3. et 4. a Success message is received by the Server ("2.04" Changed) related to the WRITE-ATTRIBUTES operation
3. In test step 5, the Success Message ("2.05" Content) is received by the Server along with the payload related to the DISCOVER request and containing:

</3/0/11>;dim=1;pmin=10;pmax=320,</3/0/11/0>;pmax=100&epmin=1&epmax=20

6.3.33. LightweightM2M-1.2-int-264 -- Discover Command on Object ID with Depth 1

Test Case Id

LightweightM2M-1.2-int-264

Test Object

Client and Server

Test Case Description

Discovering the instances of Device Object as well as reporting the various Attributes attached to Object / Object Instance

Preconditions

- The Client supports the Configuration C.4 as defined in Annex C with default pmin and pmax resources set to 1 and

10 respectively in the server object

- The Client is registered with the Server
- No <NOTIFICATION> Attribute is initially attached to the Object Device

Test Procedure

Various DISCOVER operations are performed by the Server on the Object Device. A WRITE-ATTRIBUTES command is interleaved to modify the content of the DISCOVER command reply

Normal flow:

1. The initial DISCOVER (CoAP GET Accept:40) operation is performed on Objet Device ID:3 with depth parameter set to 1
2. A WRITE-ATTRIBUTES operation set the pmax=200 <NOTIFICATION> Attributes at the Object instance level
3. DISCOVER command is performed at Object level of Object ID:3 with depth parameter set to 1

Pass-Criteria

1. In test step 1, the Success Message ("2.05" Content) is received by the Server along with the payload related to the DISCOVER request and containing :

</3>;pmin=1;pmax=10,</3/0>

2. In test step 2, a Success message is received by the Server ("2.04" Changed) related to the WRITE-ATTRIBUTES operation
3. In test step 3, the Success message ("2.05" Content) is received by the Server along with the payload related to the DISCOVER request and containing :

</3>;pmin=1;pmax=10,</3/0>;pmax=200

6.3.34. LightweightM2M-1.2-int-266 -- Discover Command Object ID with Depth 3

Test Case Id

LightweightM2M-1.2-int-266

Test Object

Client and Server

Test Case Description

Discovering the implemented Resources of the Device Object as well as reporting the various Attributes attached to Object / Object Instance / Resources and Resoirces Instances

Preconditions

- The Client supports the Configuration C.4 as defined in Annex C
- The Client is registered with the Server

- No <NOTIFICATION> Attribute is initially attached to the Object Device

Test Procedure

Various DISCOVER operations are performed by the Server on the Object Device. Several WRITE-ATTRIBUTES commands are interleaved to modify the content of the DISCOVER command replies

Normal flow:

1. The initial DISCOVER (CoAP GET Accept:40) operation is performed on Objet Device ID:3 with depth parameter set to 2
2. A WRITE-ATTRIBUTES operation set the pmin=10 & pmax=200 <NOTIFICATION> Attributes at the Object level
3. The WRITE-ATTRIBUTES operation set the lt=1, gt=6 et st=1 <NOTIFICATION> Attributes of the Resource ID:7 (Power Source Voltage)
4. The WRITE-ATTRIBUTES operation set gt=8 <NOTIFICATION> Attributes of the Resource instance ID:7/1 (External Power Source Voltage)
5. DISCOVER (CoAP GET Accept:40) operation is performed on Objet Device ID:3 with depth parameter set to 3

Pass-Criteria

1. In test step 1, the Success Message ("2.05" Content) is received by the Server along with the payload related to the DISCOVER request and containing :

< /3>;pmin=1;pmax=10,</3/0>,</3/0/1>, </3/0/2>, </3/0/3>, </3/0/4>,</3/0/6>;dim=2,</3/0/7>;dim=2, </3/0/8>;dim=2,</3/0/9>,</3/0/11>,</3/0/16>

2. In test step 2, a Success message is received by the Server ("2.04" Changed) related to the WRITE-ATTRIBUTES operation
3. In test step 3, a "Success" message is received by the Server ("2.04" Changed) related to the WRITE-ATTRIBUTES operation
4. In test step 4, a "Success" message is received by the Server ("2.04" Changed) related to the WRITE-ATTRIBUTES operation
5. In test step 5, the Success message ("2.05" Content) is received by the Server with the following payload:

</3/0>;pmin=10;pmax=200,</3/0/1>, </3/0/2>, </3/0/3>, </3/0/4>, </3/0/6>;dim=2, </3/0/6/0>, </3/0/6/1>, </3/0/7>;dim=2;lt=1;gt=6;st=1,</3/0/7/0>, </3/0/7/1>; gt=8, </3/0/8>;dim=2,</3/0/8/0>, </3/0/8/1>, </3/0/9>, </3/0/11>,</3/0/16>

6.3.35. LightweightM2M-1.1-int-270 -- Create Object Instance

As this functionality is not simply available through usage of Core Objects, the result of such Test Case is delegated to the result of any Test Case belonging to the following list:

- LightweightM2M-1.1-int-1630 -- Create Portfolio Object Instance

6.3.36. LightweightM2M-1.1-int-271 -- Create Multiple Resource Instance

6.3.37. LightweightM2M-1.1-int-280 -- Successful Read-Composite Operation

Test Case Id

LightweightM2M-1.1-int-280

Test Object

Client and Server

Test Case Description

Querying the Binding and first Error Code from the Device Object (ID: 3), and all the information of the first instance of the Server Object (ID: 1).

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server using LwM2M v1.1
- Client is supporting the Read-Composite operation

Test Procedure

A Read-Composite operation has been received by the Client from the Server on:

- Binding from the Device Object (/3/0/16)
- First Error Code from the Device Object (/3/0/11/0)
- First instance of the Server Object (/1/0)

Normal flow:

1. READ COMPOSITE (CoAP FETCH) operation is performed on:

- /3/0/16
- /3/0/11/0
- /1/0

Pass-Criteria

In test step 1. Client has received the CoAP FETCH command with the following URIs only:

- /3/0/16
- /3/0/11/0
- /1/0

In test step 1. Server has received the "Success" message (2.05 Content), along with the requested information in the SenML CBOR data format (112) or the SenML JSON data format (110). The expected values are:

- /3/0/11/0: integer value 0
- /3/0/16: string value "U"

- /1/0/0: integer value 1
- /1/0/1: integer value 86400
- /1/0/6: Boolean value false
- /1/0/7": string value "U"

Other resources under the URI "/1/0" MAY be present.

6.3.38. LightweightM2M-1.1-int-281 -- Partially Successful Read-Composite Operation

Test Case Id

LightweightM2M-1.1-int-281

Test Object

Client and Server

Test Case Description

Performing a Read-Composite operation on both readable and unreadable resources.

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server using LwM2M v1.1
- Client is supporting the Read-Composite operation

Test Procedure

A Read Composite operation has been received by the Client from the Server on:

- Registration Lifetime resource from first instance of Server object (/1/0/1)
- Binding resource from first instance of Server object (/1/0/7)
- Registration Update Trigger resource from first instance of Server object (/1/0/8)

Normal flow:

1. READ COMPOSITE (CoAP FETCH) operation is performed on:

- /1/0/1
- /1/0/7
- /1/0/8

Pass-Criteria

In test step 1. Client has received the CoAP FETCH command with the following URIs only:

- /1/0/1

- /1/0/7
- /1/0/8

In test step 1. Server has received the "Success" message (2.05 Content), along with two of the requested resources in the SenML CBOR data format (112) or the SenML JSON data format (110). The expected resources along with their values are:

- /1/0/1: integer value 86400
- /1/0/7": string value "U"

Note that the Registration Update Trigger (/1/0/8) is not readable and MUST not be part of the response.

6.3.39. LightweightM2M-1.1-int-290 -- Delete Object Instance

As this functionality is not simply available through usage of Core Objects, the result of such Test Case is delegated to the result of any Test Case belonging to the following list:

- LightweightM2M-1.1-int-1635 -- Delete Portfolio Object Instance

6.4. Information Reporting Interface [300-399]

6.4.1. LightweightM2M-1.1-int-301 -- Observation and Notification of parameter values

Test Case Id

LightweightM2M-1.1-int-301

Test Object

Client and Server

Test Case Description

Sending the observation policy to the device.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.4 as defined in Annex C
- Device is switched on and operational.
- Client is registered with the LwM2M Server

Test Procedure

The Server establishes an Observation relationship with the Client to acquire conditional notifications about Resources from the Device Object Instance:

- Power Source Voltage (ID:7)
- Power Source Current (ID:8)

Normal flow:

1. Server Determines which Power Sources are on the Device (READ access to Resource ID:6 of Device Object ID:3)
2. Server communicates with the Device Object ID:3 via WRITE-ATTRIBUTE (CoAP PUT) operations, to set the pmin & pmax Attributes of each targeted Resources (e.g. /3/0/7?pmin=5&pmax=15 and /3/0/8?pmin10&pmax=20)
3. Server sends OBSERVE (CoAP GET operation with Observe Option set to 0) messages for the targeted Resources (ID:7 & ID:8) to activate reporting
4. Client reports requested information with a NOTIFY message (CoAP responses)

Pass-Criteria

1. The Server regularly receives the requested information and displays "Power Source Voltage" and "Power Source Current" values to the user if possible.

6.4.2. LightweightM2M-1.1-int-302 -- Cancel Observations using Reset Operation

Test Case Id

LightweightM2M-1.1-int-302

Test Object

Client and Server

Test Case Description

Cancel the Observation relationship by sending "Cancel Observation" operation.

Tool

n/a

Test code

n/a

Preconditions

- The client supports the Configuration C.4 as defined in Annex C
- Device is switched on and operational.
- Client is registered at the LwM2M Server
- Server established a successful Observation relationship with the Client as defined in Test Case Id: LightweightM2M-1.1-int-301

Test Procedure

The Server removes a pre-established Observation relationship by sending "Cancel Observation". The Client removes conditional notifications about:

- Power Source Voltage
- Power Source Current

Normal flow:

1. Client reports requested information with NOTIFY messages (CoAP responses) on Resource ID:7 and ID:8
2. On receving a NOTIFY message from Resource ID:7, the Server sends a Cancel Observation (CoAP RESET message) to remove the Observation relationship with that Resource .
3. On receving a NOTIFY message from Resource ID:8, the Server sends a Cancel Observation (CoAP RESET message) to remove the Observation relationship with that Resource .
4. Client stops reporting requested information and removes associated entries from the list of observers.

Pass-Criteria

1. The Server receives regular NOTIFICATION from Device Object ID:3 on Resources Power Source Voltage (ID:7) and Power Source Current (ID:8)
2. The Server stops receiving information first on "Power Source Voltage" then on "Power Source Current" after having sent the Cancel Observation (Reset Operation) on that Resources. Associated entries from the list of observers are removed.

6.4.3. LightweightM2M-1.1-int-303 -- Cancel observations using Observe with Cancel parameter

Test Case Id

LightweightM2M-1.1-int-303

Test Object

Client and Server

Test Case Description

Cancel the Observation relationship by sending an OBSERVE operation with the CANCEL parameter

Tool

n/a

Test code

n/a

Preconditions

- The client supports the Configuration C4 as defined in Annex C
- Device is switched on and operational.
- Client is registered at the LwM2M Server
- Server established a successful Observation relationship with the Client as defined in Test Case Id: LightweightM2M-1.1-int-301

Test Procedure

The Server removes a pre-established Observation relationship by sending "Observe" with Cancel Parameter. The Client removes conditional notifications about:

- Power Source Voltage
- Power Source Current

Normal flow:

1. Client reports requested information with a NOTIFY message (CoAP responses)
2. After receiving few notifications, Server sends OBSERVE operation with the CANCEL parameter (CoAP GET message with Observe option set to 1) to cancel the Observation relationship with the Instance of the Device Object (/3/0)
3. Client stops reporting requested information on both Resources and removes associated entries from the list of observers.

Pass-Criteria

1. The Server receives regular NOTIFICATION from Device Object ID:3 on Resources Power Source Voltage (ID:7) and Power Source Current (ID:8)
2. The Server stops receiving information on "Power Source Voltage" and "Power Source Current" after having sent the Cancel Observation (with Cancel parameter) on the Device Object Instance (/3/0).Associated entries from the list of observers are removed.

6.4.4. LightweightM2M-1.1-int-304 -- Observe-Composite Operation

Test Case Id

LightweightM2M-1.1-int-304

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with Observe-Composite operation.

Note: support of this operation is not mandatory Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports mandatory resources of LwM2M Server and Device Objects
- No observation attributes are set and no Observation is set on Client

Test Procedure

1. Server sets attributes on Lifetime Resource (/1/x/1, where x in a LwM2M Server Object Instance) with Write-Attributes (CoAP PUT) operation:

- pmin=30
- pmax=45

2. Server performs an Observe-Composite (CoAP FETCH with Observe option = 0) on the following combination of Resources and Resources Instances:

- Lifetime Resource (/1/x/1, where x in a LwM2M Server Object Instance),
- Error Code Resource Instance (/3/0/11/y, where y in an existing Error Code Resource Instance),
- Supported Binding and Modes Resource of Device Object (/3/0/16)

"Content Format ID" is set to either SenML CBOR or SenML JSON

"Accept: Content-format" is set either to SenML CBOR or SenML JSON

3. Wait for 45 seconds to retrieve a Notification containing value of all of the Observed resources.

Pass-Criteria

1. Client accepted the request and replied with 2.04 Changed.
2. Client accepted the request and replied with 2.05 Content (Observe Response), containing the values of all observed Resources.
3. Composite Notification was sent at least 30 seconds after requesting it, but not later than 45 sec. Notification has a format of CoAP 2.05 Content with Observe = 1.\

"Content-format" is set either to SenML CBOR or SenML JSON.\

Message payload contains values of all of the observed resources from step 2.

6.4.5. LightweightM2M-1.1-int-305 -- Cancel Observation-Composite Operation

Test Case Id

LightweightM2M-1.1-int-305

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with Cancel Observation-Composite operation.

Note: support of this operation is not mandatory

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports mandatory resources of LwM2M Server and Device Objects
- No observation attributes are set and no Observation is set on Client

Test Procedure

1. Server performs an Observe-Composite (CoAP FETCH with Observe option = 0) on the following combination of Resources and Resources Instances:

- Lifetime Resource (/1/x/1, where x in a LwM2M Server Object Instance),
- Error Code Resource Instance (/3/0/11/y, where y in an existing Error Code Resource Instance),
- Supported Binding and Modes Resource of Device Object (/3/0/16)

"Content Format ID" is set to either SenML CBOR or SenML JSON

"Accept: Content-format" is set either to SenML CBOR or SenML JSON

2. Server performs a Cancel Observation-Composite (CoAP FETCH with Observe option = 1) on the same combination of Resources and Resources Instances:

- Lifetime Resource (/1/x/1, where x in a LwM2M Server Object Instance),
- Error Code Resource Instance (/3/0/11/y, where y in an existing Error Code Resource Instance),
- Supported Binding and Modes Resource of Device Object (/3/0/16)

"Content Format ID" is set to either SenML CBOR or SenML JSON

Pass-Criteria

1. Client accepted the request and replied with 2.05 Content (Observe Response), containing the values of all observed Resources.
2. Client accepted the request and replied with 2.05 Content (Observe Response), containing the values of all observed Resources. No notification arrives after this point

6.4.6. LightweightM2M-1.1-int-306 -- Send Operation

Test Case Id

LightweightM2M-1.1-int-306

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with Send operation.

Note: support of this operation is not mandatory

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports mandatory resources of LwM2M Server and Device Objects
- Client has a Send mechanism configured on any Resource/Resources Instances configured. Sends are sent periodically with small interval.

Test Procedure

1. Wait for (or enforce) Client to perform Send (CoAP POST on /dp URI) operation with value of any Resource/Resource Instances.

Pass-Criteria

1. Send operation MUST contain Resources/Resources Instances of Objects introduced during Registration.\

   "Content Format" is set to either SenML CBOR (112) or SenML JSON (110).\

   Operation is accepted by the Server. Server replied with 2.04 Changed response.

6.4.7. LightweightM2M-1.1-int-307 -- Muting Send

Test Case Id

LightweightM2M-1.1-int-307

Test Object

Client and Server

Test Case Description

Purpose of this test is to show conformance with Mute Send functionality\

Note: support of this operation is not mandatory

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports mandatory resources of LwM2M Server and Device Objects
- Client has a Send mechanism configured on any Resource/Resources Instances configured. Sends are sent periodically with small interval.
- Non-mandatory Mute Send Resource is present

Test Procedure

1. Wait for (or enforce) Client to perform Send (CoAP POST on /dp URI) operation with value of any Resource/Resource Instances, to confirm that mechanism is up and running
2. Write 'true' value into Mute Send Resource (/1/x/23, where x in a LwM2M Server Object Instance) with Write operation (CoAP POST)
3. Wait for some time of try triggering Send operation on Client
4. Write 'false' value into Mute Send Resource (/1/x/23) with Write operation (CoAP PUT)
5. Wait for some time of try triggering Send operation on Client

Pass-Criteria

1. Send operation MUST contain Resources/Resources Instances of Objects introduced during Registration.\

   "Content Format" is set to either SenML CBOR (112) or SenML JSON (110).\

   Operation is accepted by the Server. Server replied with 2.04 Changed response.
2. Client accepted such message and replies with 2.04 Changed response.

3. Even tough Send is triggered - it is not sent towards Server
4. Client accepted such message and replied with 2.04 Changed response.
5. Send (CoAP POST on /dp URI) is sent towards a Server

6.4.8. LightweightM2M-1.1-int-308 -- Observe-Composite and Creating Object Instance

Note: this functionality is not simply available through usage of Core Objects. Client needs to support non-mandatory Portfolio Object.

Test Case Id

LightweightM2M-1.1-int-308

Test Object

Client and Server

Test Case Description

Purpose of this test is to check if Observe-Composite values of newly created Object Instance will be included in Notifications.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports mandatory resources of LwM2M Server and Device Objects
- No observation attributes are set and no Observation is set on Client
- Client supports non-mandatory Portfolio Object

Test Procedure

1. Server initializes Portfolio Object Instance 0 (if missing on Device), by performing Create (CoAP POST) on: Portfolio Object, Instance 0, resource Identity (ID: 0):

    1. Identity.0 = aa (/16/0/0/0),
    2. Identity.1 = bb (/16/0/0/1),
    3. Identity.2 = cc (/16/0/0/2),
    4. Identity.3 = dd (/16/0/0/3),

    Portfolio Object Instance 1 MUST not exist on a Client.

2. Server Writes Attributes on newly created Object Instance (/16/0) with Write-Attributes (CoAP PUT) operation:

- pmin=30
- pmax=45

3. Server performs an Observe-Composite (CoAP FETCH with Observe option = 0) on the following combination of Object Instances:

- Portfolio.0 (/16/0)
- Portfolio.1 (/16/1)

"Content Format ID" is set to either SenML CBOR or SenML JSON

"Accept: Content-format" is set either to SenML CBOR or SenML JSON

4. Wait for 45 seconds to retrieve a Notification
5. Server initializes Portfolio Object Instance 1 by performing Create (CoAP POST) on: Portfolio Object, Instance 1, resource Identity (ID: 0):

```
1. Identity.0 = 11 (/16/0/0/0),
2. Identity.1 = 22 (/16/0/0/1),
3. Identity.2 = 33 (/16/0/0/2),
4. Identity.3 = 44 (/16/0/0/3),
```

6. Wait for 45 seconds to retrieve a Notification

Pass-Criteria

1. Client accepted the request and replied with 2.01 Created.
2. Client accepted the request and replied with 2.04 Changed.
3. Client accepted the request and replied with 2.05 Content (Observe Response), containing the values of Portfolio Object Instance 0 only (as Instance 1 does not exist)
4. Composite Notification was sent at least 30 seconds after requesting it, but not later than 45 seconds. Notification has a format of CoAP 2.05 Content with Observe = 1. Notification contains the value of Portfolio Object Instance 0 only (as Instance 1 does not exist)\

"Content-format" is set either to SenML CBOR or SenML JSON.

5. Client accepted the request and replied with 2.01 Created.
6. Composite Notification arrived at least 30 seconds after the previous one, but before 45 seconds has passed. Notification has a format of CoAP 2.05 Content with Observe = 1. Notification contains the value of Portfolio Object Instance 0 and Instance 1 (newly created)\

"Content-format" is set either to SenML CBOR or SenML JSON.

6.4.9. LightweightM2M-1.1-int-309 -- Observe-Composite and Deleting Object Instance

Note: this functionality is not simply available through usage of Core Objects. Client needs to support non-mandatory Portfolio Object.

Test Case Id

LightweightM2M-1.1-int-309

Test Object

Client and Server

Test Case Description

Purpose of this test is to check if Observe-Composite Notification reflect the

change when deleting Observed Object Instance. Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server
- Client supports mandatory resources of LwM2M Server and Device Objects
- No observation attributes are set and no Observation is set on Client

Test Procedure

1. Server initializes Portfolio Object Instance 0 (if missing on Device), by performing Create (CoAP POST) on: Portfolio Object, Instance 0, resource Identity (ID: 0):

a. Identity.0 = aa (/16/0/0/0),

b. Identity.1 = bb (/16/0/0/1),

c. Identity.2 = cc (/16/0/0/2),

d. Identity.3 = dd (/16/0/0/3),

2. Server initializes Portfolio Object Instance 1 by performing Create (CoAP POST) on: Portfolio Object, Instance 1, resource Identity (ID: 0):

a. Identity.0 = 11 (/16/1/0/0),

b. Identity.1 = 22 (/16/1/0/1),

c. Identity.2 = 33 (/16/1/0/2),

d. Identity.3 = 44 (/16/1/0/3),

3. Server Writes Attributes on Portfolio Object Instance 0 (/16/0) with Write-Attributes (CoAP PUT) operation:

- pmin=30
- pmax=45

4. Server performs an Observe-Composite (CoAP FETCH with Observe option = 0) on the following combination of Object Instances:

- Portfolio.0 (/16/0)
- Portfolio.1 (/16/1)

"Content Format ID" is set to either SenML CBOR or SenML JSON

"Accept: Content-format" is set either to SenML CBOR or SenML JSON

5. Wait for 45 seconds to retrieve a Notification
6. Server deletes Portfolio Object Instance 1 by performing Delete (CoAP DELETE) on: Portfolio Object, Instance 1
7. Wait for 45 seconds to retrieve a Notification

Pass-Criteria

1. Client accepted the request and replied with 2.01 Created.
2. Client accepted the request and replied with 2.01 Created.
3. Client accepted the request and replied with 2.04 Changed.
4. Client accepted the request and replied with 2.05 Content (Observe Response), containing the values of Portfolio Object Instance 0 and Instance 1. "Content-format" is set either to SenML CBOR or SenML JSON.
5. Composite Notification arrived at least 30 seconds after ordering it, but before 45 seconds has passed. Notification has a format of CoAP 2.05 Content with Observe = 1. Notification contains the value of Portfolio Object Instance 0 and Instance 1. "Content-format" is set either to SenML CBOR or SenML JSON.
6. Client accepted the request and replied with 2.04 Deleted.
7. Composite Notification arrived at least 30 seconds after the previous one, but before 45 seconds has passed. Notification has a format of CoAP 2.05 Content with Observe = 1. Notification contains the value of Portfolio Object Instance 0 only (as Instance 1 was deleted).\

"Content-format" is set either to SenML CBOR or SenML JSON.

6.4.10. LightweightM2M-1.1-int-310 -- Observe-Composite and modification of parameter values

Test Case Id

LightweightM2M-1.1-int-310

Test Object

Client and Server

Test Case Description

Sending the observation policy to the Lifetime of Server object instance (ID: 1) and Device object (ID: 3)

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server in LwM2M 1.1
- Client and Server MUST support the Observe-Composite operation

Test Procedure

The Server sends an Observe-Composite (CoAP FETCH) operation to the Client on:

- Lifetime resource (ID: 3) from instance 0 of Server object (ID: 1)
- Device object (ID: 3)

Then the Server sends a Write-Attributes operation to the Client on Device object (ID: 3):

- pmax=5

Normal flow:

1. An OBSERVE-COMPOSITE (CoAP FETCH operation with Observe Option set to 1) operation is performed on /1/0/3 and /3/0
2. Client reports requested information with NOTIFY message (CoAP response)
3. A WRITE-ATTRIBUTE operation sets the pmax <NOTIFICATION> Attribute to 5 seconds on the Device object (ID: 3)
4. Client reports requested information with NOTIFY message (CoAP response) each 5 seconds.

Pass-Criteria

1. In test step 1. Server sends a CoAP FETCH command with only the URIs /1/0/3 and /3. Then it receives a "success" message (2.05 Content) containing the requested Lifetime of Server object instance (ID: 1) and Device object (ID: 3) values from the Client.
2. In test step 2. Client reports requested information with a NOTIFY message (CoAP response) with notification number to 0
3. In test step 3. a Success message is received by the Server ("2.04" Changed) related to the WRITE-ATTRIBUTES operation
4. In test step 4. the Server regularly receives the requested Lifetime of Server object instance (ID: 1) and Device object (ID: 3) values from the Client.

6.4.11. LightweightM2M-1.1-int-311 -- Send command

Test Case Id

LightweightM2M-1.1-int-311

Test Object

Client and Server

Test Case Description

Send Information of Lifetime resource of Server object instance (ID: 1) and Error code resource of Device object (ID: 3)

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- Client is registered with the LwM2M Server in LwM2M 1.1
- Client and Server MUST support the Send operation

Test Procedure

A Send operation has been received by the Server from the Client on:

- Lifetime resource from first instance of Server object (/1/0/1)
- Error Code resource of Device object (/3/0/11)

Normal flow:

1. A SEND (CoAP POST with /dp as path) operation is performed by the Client with the following information:

- Lifetime (/1/0/1): 86400
- Error Code (/3/0/11/0): 0

Pass-Criteria

1. In test step 1. the Server received the SEND message (CoAP POST with /dp as path), along with the Client information and containing the expected values defined for this test
2. In test step 1. the Client receives a "Success" message (2.04 Changed) for SEND operation

6.4.12. LightweightM2M-1.2-int-312 -- Observe operation carrying observe attributes as parameterss

Test Case Id

LightweightM2M-1.2-int-312

Test Object

Client and Server

Test Case Description

Sending the observation policy to the device.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports Configuration C.4 as defined in Annex C
- Device is switched on and operational.
- Client is registered with the LwM2M Server

Test Procedure

The Server establishes an Observation relationship with the Client to acquire conditional notifications about Resources from the Device Object Instance:

- Power Source Voltage (ID:7)

Normal flow:

1. Server sends OBSERVE (CoAP GET /3/0/7 ?pmin=5 &pmax=15 with Observe Option set to 0) messages for Resources ID:7 to activate reporting
2. Client reports requested information with a NOTIFY message (CoAP responses)

Pass-Criteria

1. In step 1 server receives voltage values for Internal and External battery as represented by /3/0/7/0 and /3/0/7/1
2. Subsequently the Server receives Internal and External battery voltage levels at least every 15 seconds

6.4.13. LightweightM2M-1.2-int-313 -- Observation with Notification parameter values overeiding existing attributes

Test Case Id

LightweightM2M-1.2-int-313

Test Object

Client and Server

Test Case Description

Sending the observation policy to the device.

Tool

n/a

Test code

n/a

Preconditions

- The Client supports Configuration C.4 as defined in Annex C
- Device is switched on and operational.
- Client is registered with the LwM2M Server

Test Procedure

The Server establishes an Observation relationship with the Client to acquire conditional notifications about Resources from the Device Object Instance:

- Power Source Voltage (ID:7)

Normal flow:

1. Server communicates with the Device Object ID:3 via WRITE-ATTRIBUTE operations, to set the pmin & pmax Attributes of Resource 7 (e.g. CoAP PUT /3/0/7?pmin=5&pmax=60
2. Server sends OBSERVE (CoAP GET /3/0/7 ?pmin=5 &pmax=10 with Observe Option set to 0) messages for Resources ID:7 to activate reporting
3. Client reports requested information with a NOTIFY message (CoAP responses)

Pass-Criteria

1.In test step 1, a Success message is received by the Server ("2.04" Changed) related to the WRITE-ATTRIBUTES operation

2.In step 2 server receives voltage values for Internal and External battery as represented by /3/0/7/0 and /3/0/7/1

3. Subsequently the Server receives Internal and External battery voltage levels at least every 10 seconds

6.5. Security [400-499]

6.5.1. LightweightM2M-1.1-int-401 -- UDP Channel Security -- Pre-shared Key Mode

Test Case Id

LightweightM2M-1.1-int-401

Test Object

Client and Server

Test Case Description

Establishing a successful DTLS session using UDP pre-shared key mode

Tool

n/a

Test code

n/a

Preconditions

- The client supports the Configuration C.1 as defined in Annex C
- The bootstrap procedure has been completed or the required bootstrap information is available to the client.
- The bootstrap information includes the Security Mode resource of the object LwM2M Security set to 0: Pre-Shared Key mode

Test Procedure

Device is switched on.

A DTLS session is established between client and Server and the Client automatically registers with that Server

Using this DTLS session, a simple data exchange is performed between the Server and the Client, to verify the encrypted connection is properly established.

Normal flow:

1. DTLS Session is established
2. Registration message (CoAP POST) is sent from LwM2M Client to LwM2M Server.
3. Client receives Success message (2.01 Created) from the Server.
4. READ (CoAP GET) on the Instance of the Device Object is performed using the default TLV data format (cf Test LwM2M-1.1-int-203)
5. Server receives success message (2.05 Content) and the requested values (encrypted)

Pass-Criteria

1. In test step 2 & 3, Registration command of the Client on the Server is performed successfully over the DTLS session
2. In test step 4 & 5 the READ command work successfully over the DTLS session.

6.5.2. LightweightM2M-1.0-int-402 -- UDP Channel Security -- Certificate Mode

Test Case Id

LightweightM2M-1.1-int-402

Test Object

Client and Server

Test Case Description

Establishing a successful DTLS session using certificate mode

Tool

n/a

Test code

n/a

Preconditions

- The client supports the Configuration C.17 as defined in Annex C
- The bootstrap procedure has been completed or the required bootstrap information is available to the client.
- DNS is configured in such a way that the server responds to server.example.com.

Test Procedure

Device is switched on.

A DTLS session is established between client and Server and the Client automatically registers with that Server

Using this DTLS session, a simple data exchange is performed between the Server and the Client, to verify the encrypted connection is properly established.

Normal flow:

1. DTLS Session is established
2. Registration message (CoAP POST) is sent from LwM2M Client to LwM2M Server.
3. Client receives Success message (2.01 Created) from the Server.
4. READ (CoAP GET) on the Instance of the Device Object is performed using the default TLV data format (cf Test LwM2M-1.1-int-203)
5. Server receives success message (2.05 Content) and the requested values (encrypted)

Pass-Criteria

1. In test step 2 & 3, Registration command of the Client on the Server is performed successfully over the DTLS session
2. In test step 4 & 5 the READ command works successfully over the DTLS session.

6.5.3. LightweightM2M-1.1-int-403 -- UDP Channel Security -- Certificate Mode (Server Identify Verification Failure)

Test Case Id

LightweightM2M-1.1-int-403

Test Object

Client and Server

Test Case Description

Failure to establish a DTLS session using certificate mode because of Server Identity Verification Failure

Tool

n/a

Test code

n/a

Preconditions

- The client supports the Configuration C.18 as defined in Annex C
- The bootstrap procedure has been completed or the required bootstrap information is available to the client.
- DNS is configured in such a way that the server responds to server-fail.example.com.

Test Procedure

Device is switched on.

A DTLS handshake is started between the DTLS Client and the DTLS Server. The DTLS handshake fails because the content of the certificate (server.example.com) does not match the configured LwM2M Server URI (server-fail.example.com).

Pass-Criteria

The DTLS Handshake fails

6.5.4. LightweightM2M-1.1-int-404 -- TCP Channel Security -- Certificate Mode

Test Case Id

LightweightM2M-1.1-int-404

Test Object

LwM2M Client and LwM2M Server

Test Case Description

Establishing a successful TLS session over TCP using the certificate mode

Tool

n/a

Test code

n/a

Preconditions

- Configuration C.19, as defined in Annex C, is available to the LwM2M Client. The Security Mode Resource of the LwM2M Security Object is set to 2 (" Certificate mode").

Test Procedure

Device is switched on.

A TLS handshake using certificate-based authentication is executed between the LwM2M Client and the LwM2M Bootstrap-Server using CoAP over TLS over TCP. Once the bootstrap procedure is complete, the LwM2M Client automatically registers with the LwM2M Server over TLS over TCP.

Successful bootstrapping and the subsequent registration by the LwM2M Client at the LwM2M Server is an indication that the use of LwM2M using CoAP over TLS over TCP was successful.

Normal flow:

1. The LwM2M Client initiates a TCP connection with the LwM2M Bootstrap-Server.
2. The LwM2M Client initiates a TLS handshake with the LwM2M Bootstrap-Server.
3. CSM (Capabilities and Settings Messages) are exchanged between two endpoints (LwM2M Client and LwM2M Bootstrap-Server) according to the CoAP over TCP/TLS specification.
4. The LwM2M bootstrapping occurs and the LwM2M Client gets configured with information about the LwM2M Server.
5. The LwM2M Client initiates a TCP connection with the LwM2M Server.
6. The LwM2M Client initiates a TLS handshake with the LwM2M Server.
7. The LwM2M Client performs the LwM2M registration step with the LwM2M Server.
8. Registration message (CoAP POST) is sent from LwM2M Client to LwM2M Server.
9. Server performs a READ operation (CoAP GET) on the Instance of the Device Object.

Pass-Criteria

1. A TCP session is established successfully.
2. The TLS handshake is successfully established with certificate-based credentials.
3. CSM messages are exchanged and accepted by both parties
4. Client sent REGISTER message (CoAP POST on /rd with appropriate query) towards Server along with the following information:

- Endpoint Client Name (optional)

- Registration lifetime
- LwM2M version
- Binding mode = T
- SMS number (optional)
- Objects and Object Instances (mandatory and optional objects / object instances) ; possibly with Version of Objects. Payload type of Object list SHOULD be CoRE Link Format (application/link-format), for example: </1/2>,</2>, </3/0>,</4/0>,</5/0>,</6/0>,</7/0>,</10>;ver="1. **Note**: Object /0 and /21 MUST not be part of this list. Server replied with Success message (CoAP 2.01 Created).
- Client replied with CoAP 2.05 Content. Message payload contains Object Instance and all of the Object Resources available on Client. Message Content-type is chosen by Client. Reply is transferred over the secured TLS communication channel.

6.5.5. LightweightM2M-1.1-int-405 -- OSCORE Security

Test Case Id

LightweightM2M-1.1-int-405

Test Object

Client and Server

Test Case Description

Establishing a successful LwM2M session using OSCORE security.

Tool

n/a

Test code

n/a

Preconditions

- The client supports the Configuration C.19 as defined in Annex C
- The bootstrap procedure has been completed or the required bootstrap information is available to the client.

Test Procedure

Device is switched on.

A communication is established between Client and Server and the Client automatically registers with that Server.

A simple data exchange is performed between the Server and the Client, to verify the exchanged CoAP messages are correctly encrypted using OSCORE.

Normal flow:

1. Client initiates UDP or TCP Connection
2. OSCORE Request (CoAP POST) containing the Registration message is sent from LwM2M Client to LwM2M Server.
3. Server performs READ (CoAP GET) on the Instance of the Device Object

Pass-Criteria

1. ll CoAP messages exchanged between the Client and the Server have the OSCORE Option, their code is either POST or 2.04 Changed, their payload is encrypted.
2. Client is registered to the Server with the following information:

- Endpoint Client Name (optional)
- registration lifetime
- LwM2M version
- binding mode
- SMS number (optional)
- Objects and Object Instances (mandatory and optional objects / object instances) ; possibly with Version of Objects. Payload type of Object list SHOULD be CoRE Link Format (application/link-format), for example: </1/2>,</2>, </3/0>,</4/0>,</5/0>,</6/0>,</7/0>,</10>;ver="1.

Note: Object /0 and /21 MUST not be part of this list

Server replied with Success message (CoAP 2.01 Created).

3. Client replied to the Read Operation with the content of the Device Object Instance and all of the Device Object Resources available on Client.

6.5.6. LightweightM2M-1.1-int-406 -- TCP Channel Security -- Pre-shared Key Mode

Test Case Id

LightweightM2M-1.1-int-406

Test Object

Client and Server

Test Case Description

Establishing a successful TLS session using TCP - Pre-shared Key Mode

Tool

n/a

Test code

n/a

Preconditions

- Configuration C.20, as defined in Annex C, is available to the LwM2M Client. The Security Mode Resource of the LwM2M Security Object is set to 0 (" Pre-Shared Key mode").

Test Procedure

Device is switched on.

A TLS session is established between the LwM2M Client and the LwM2M Bootstrap-Server using CoAP over TLS over TCP. Once the bootstrap procedure is complete, the LwM2M Client automatically registers with the LwM2M Server over TLS over TCP.

Successful bootstrapping and the subsequent registration by the LwM2M Client at the LwM2M Server is an indication that the use of LwM2M using CoAP over TLS over TCP was successful.

Normal flow:

1. The LwM2M Client initiates a TCP connection with the LwM2M Bootstrap-Server.
2. The LwM2M Client initiates a TLS handshake with the LwM2M Bootstrap-Server.
3. CSM (Capabilities and Settings Messages) are exchanged between two endpoints (LwM2M Client and LwM2M Bootstrap-Server) according to the CoAP over TCP/TLS specification.
4. The LwM2M bootstrapping occurs and the LwM2M Client gets configured with information about the LwM2M Server.
5. The LwM2M Client initiates a TCP connection with the LwM2M Server.
6. The LwM2M Client initiates a TLS handshake with the LwM2M Server.
7. The LwM2M Client performs the LwM2M registration step with the LwM2M Server.
8. Registration message (CoAP POST) is sent from LwM2M Client to LwM2M Server.
9. Server performs a READ operation (CoAP GET) on the Instance of the Device Object.

Pass-Criteria

1. A TCP session is established successfully.
2. The TLS handshake is successfully established with pre-shared secret credentials.
3. CSM messages are exchanged and accepted by both parties.
4. The LwM2M Client sent REGISTER message (CoAP POST on /rd with appropriate query) towards LwM2M Server along with the following information:

- Endpoint Client Name (optional)
- Registration lifetime
- LwM2M version
- Binding mode = T

- SMS number (optional)
- Objects and Object Instances (mandatory and optional objects / object instances) ; possibly with Version of Objects. Payload type of Object list SHOULD be CoRE Link Format (application/link-format), for example: </1/2>,</2>, </3/0>,</4/0>,</5/0>,</6/0>,</7/0>,</10>;ver="1. **Note**: Object /0 and /21 MUST not be part of this list. Server replied with Success message (CoAP 2.01 Created).
- Client replied with CoAP 2.05 Content. Message payload contains Object Instance and all of the Object Resources available on Client. Message Content-type is chosen by Client. Reply is transferred over the secured TLS communication channel.

6.6. Core Specific Objects Test cases [500-999]

6.6.1. Security Object (ID 0) [500-549]

6.6.2. Server Object (ID 1) [550-599]

6.6.2.1. LightweightM2M-1.1-int-551 -- Access Check to the Resources

LightweightM2M-1.1-int-215 Test Case (Setting basic information in TLV format) is covering the same functional scope than this LightweightM2M-1.1-int-551 Test Case. The Test Status of both Test Cases MUST be considered as identical.

6.6.2.2. LightweightM2M-1.1-int-555 -- Check of the Disable (De-Registration) capability

<Test Case to fill-up>

Note : LightweightM2M-1.1-int-103 Test Case (De-Registration) is partially addressing the scope of this LightweightM2M-1.1-int-555 Test Case.

6.6.2.3. LightweightM2M-1.1-int-556 -- Check of the Update Registration capability

LightweightM2M-1.1-int-104 Test Case (Registration Update Trigger) is covering the same functional scope than this LightweightM2M-1.1-int-556 Test Case. The Test Status of both Test Cases MUST be considered as identical.

paragraph

6.6.2.4. LightweightM2M-1.1-int-560 -- Delayed Report Notification

<Test Cases to fill-up>

6.6.2.5. LightweightM2M-1.1-int-565 -- Create Object Instance

<Test Cases to fill-up>

6.6.2.6. LightweightM2M-1.1-int-566 -- Delete Object Instance

<Test Cases to fill-up>

6.6.3. Access Control Object (ID 2) [600-649]

6.6.4. LwM2M Device Object (ID 3) [650-699]

6.6.4.1. LightweightM2M-1.1-int-651 -- Check Access to the Resources

Test Case Id

LightweightM2M-1.1-int-651

Test Object

Client and Server

Test Case Description

Setting writable Resources of Device Object (ID:3) Instance in using the TLV data format (11542), checking the modifications and querying the readable Resources. Then restoring the Resources to their initial values with verification

Preconditions

- The Client supports the Configuration C.6 as defined in Annex C
- The Client is registered with the Server
- The Initial values of the Device Object (ID:3) Instance, are saved on the Server

Test Procedure

A WRITE operation from Server on the Device Object (ID:3) Instance has been received by the client using TLV data format (11542). The set of values to write is given in the sample below (651-SetOfValues)

Normal flow:

1. A 1^st^ WRITE (CoAP POST) operation on the Device Object (ID:3) Instance is performed in using the set of values contains in the 651-SetOfValues sample below.The data format TLV is used. (11542)
2. The Server READs the result of the WRITE operation by querying the Device Object (ID:3) Instance.
3. A 2^nd^ WRITE (CoAP PUT) operation on the Device Object (ID:3) Instance is performed in using the Initial values which have been preserved (pre-conditions).
4. The Server READs the result of the previous WRITE operation by querying the Server Object (ID:3) Instance.

Pass-Criteria

1. The Server receives the correct status codes for the steps 1. (2.04), 2. (2.05), 3. (2.04), & 4 (2.05) of the test.
2. In test step 3., the received values are as expected (readable resources) and consistent with the 651-SetOfValues

sample.

3. In test step 7, the received values are consistent with the values present in the initial Configuration C.3

651-SetOfValues definition

lwm2m+json version

[{"bn":"/","**n":"3/0/13","v":1367491215},**

** {"n":"3/0/14","sv":"+02:00"},**

** {"n":"3/0/15","sv":"**[Europe/Paris]**"}**]

lwm2m+tlv version

C4 0D 51 82 42 8F

C6 0E 2B 30 32 3A 30 30

C8 0F 0C 45 75 72 6F 70 65 2F 50 61 72 69 73

6.6.4.2. LightweightM2M-1.1-int-652 -- Querying the firmware version from the client

Test Case Id

LightweightM2M-1.1-int-652

Test Object

Client and Server

Test Case Description

Querying the version of the firmware present in the LwM2M Client

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.6 as defined in Annex C
- The Client is registered with the Server

Test Procedure

A READ operation from Server on the Firmware Version Resource (ID:3) has been received by the Client.

Normal flow:

1. READ (CoAP GET) operation is performed on Device Object Resource "Firmware Version"

Pass-Criteria

1. In test step 2, the Server received the requested information (Firmware Version) in the data format preferred by the Client
2. In test step 2 the Server receives the success message (2.05 Content)

6.6.4.3. LightweightM2M-1.1-int-655 -- Check of the "Reboot" capability

LightweightM2M-1.1-int-241 Test Case (Executable Resource: Rebooting the Device) is covering the same functional scope than this LightweightM2M-1.1-int-655 Test Case. The Test Status of both Test Cases MUST be considered as identical.

6.6.4.4. LightweightM2M-1.1-int-656 -- Check of the "Factory Reset" capability

Note: this test SHOULD be performed only if the client is able to carry out one of the Bootstrap methods defined in [LwM2M-CORE TS 1.1] section 5.2.3

<Test Cases to fill-up>

6.6.4.5. LightweightM2M-1.1-int-657 -- Check of the "Error Code" functionality

<Test Cases to fill-up>

6.6.4.6. LightweightM2M-1.1-int-660 -- Basic Observation and notification of Device Object Resources

LightweightM2M-1.1-int-301 Test Case (Observation and Notification of parameter values) is covering the same functional scope than this LightweightM2M-1.1-int-660 Test Case. The Test Status of both Test Cases MUST be considered as identical.

6.6.4.7. LightweightM2M-1.1-int-661 -- Extended Observation and notification of Device Object Resources

<Test Cases to fill-up>

6.6.4.8. LightweightM2M-1.1-int-670 -- Create Multiple Resource Instances

<Test Cases to fill-up>

6.6.4.9. LightweightM2M-1.1-int-680 -- Create Object Instance

Note: This operation is not allowed on this Object since the Device Object is a Mandatory and Single-instance Object.

Test Case Id

LightweightM2M-1.1-int-680

Test Object

Client and Server

Test Case Description

Verifying that Creating an Instance of the Device Object is not allowed

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- The Client is registered with the Server

Test Procedure

A CREATE operation targeting the Device Object ID:3 is performed by the Server and received by the Client.

Normal flow:

1. CREATE (CoAP POST) operation is performed on Device Object

Pass-Criteria

1. In test step 1, the Server receives the failure message (4.05 Method Not Allowed)

6.6.4.10. LightweightM2M-1.1-int-685 -- Delete Object Instance

Note: This operation is not allowed on this Object since the Device Object is a Mandatory and Single-instance Object fully maintained by the lwM2M Client. It MUST be always present when the Client is registered.

Test Case Id

LightweightM2M-1.1-int-685

Test Object

Client and Server

Test Case Description

Verifying that Deleting the Instance of the Device Object is not allowed

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.1 as defined in Annex C
- The Client is registered with the Server

Test Procedure

A DELETE operation targeting the Device Object ID:3 is performed by the Server and received by the Client.

Normal flow:

1. DELETE (CoAP DELETE) operation is performed on the Device Object Instance (/3/0)

Pass-Criteria

1. In test step 1 the Server receives the failure message (4.05 Method Not Allowed)

6.6.5. LwM2M Connectivity Monitoring (ID:4) [700-749]

6.6.5.1. LightweightM2M-1.1-int-701 -- Querying the readable resources of object

Test Case Id

LightweightM2M-1.1-int-701

Test Object

Client and Server

Test Case Description

Querying the readable Resources of the Connectivity Monitoring Object Instance and verifying the received values are as expected and coherent.

Preconditions

- The Client supports the Configuration C.5 as defined in Annex C
- Client is registered with the Server

Test Procedure

A READ operation from Server on the Connectivity Monitoring Object Instance has been received by the Client.

Normal flow:

1. READ (CoAP GET) operation is performed by the Server on the Connectivity Monitoring Object Instance of the Client

Pass-Criteria

1. In test step 1., the Client received a READ (Coap GET) command from the Server on the Connectivity Monitoring Object Instance
2. In test step 1., the Server receives the status code "2.05" for READ message success
3. In test step 1., along with the success message, the mandatory Resources (ID:0, 1,2, 4) and the optional ones, are received by the Server with expected values in compliance with LwM2M technical specification TS 1.0.

6.6.5.2. LightweightM2M-1.1-int-705 -- Setting the writable resources

There is no writable resources for this object.

6.6.5.3. LightweightM2M-1.1-int-710 -- Basic Observation and notification on Connectivity Monitoring Object Resources

Test Case Id

LightweightM2M-1.1-int-710

Test Object

Client and Server

Test Case Description

Sending the observation policy to the Client for resources of Connectivity Monitoring (ID:4)

Preconditions

- The Client supports the Configuration C.5 as defined in Annex C
- Device is switched on and operational
- Client is registered with the LwM2M server

Test Procedure

The Server establishes an Observation relationship with the Client to acquire condition notifications about the Connectivity Monitoring Object Instance.

Normal flow:

1. The Server communicates to the Client the pmin=2 and pmax=10 periods, threshold values with a WRITE ATTRIBUTE (CoAP PUT) operation at the Connectivity Monitoring Object Instance level
2. The Server sends OBSERVE (CoAP Observe Option) message to activate reporting on the Monitoring Object Instance
3. The Client reports requested information with a NOTIFY message (CoAP response)

Pass-Criteria

1. In test step 1., the Client has received a WRITE ATTRIBUTE Command (CoAP PUT) targeting the Connectivity Monitoring Object Instance with the proper pmin=2, and pmax=10 parameters
2. In test step 1, the Server received the success message (2.04 Changed) in response to the WRITE ATTRIBUTE command
3. In test step 2. the Client received the OBSERVE operation targeting the Connectivity Monitoring Object Instance
4. In test step 2., in response to its OBSERVE request, the Server receives the success message (Content 2.05) along with the Connectivity Object Instance initial values
5. In test step 3., based on pmin/pmax periods parameters received in test step 1., the Client reports Information to the Server with a NOTIFY message containing the Connectivity Object Instance updated values.
6. In test step 3., the values receives by the Server along with the success message (Content 2.05) MUST be as expected

6.6.5.4. LightweightM2M-1.1-int-711 -- Extended Observation and notification of Connectivity Monitoring Object Resources

Test Cases to fill-up.

6.6.5.5. LightweightM2M-1.1-int-720 -- Create Multiple Resource Instances

Test Cases to fill-up.

6.6.5.6. LightweightM2M-1.1-int-730 -- Create Object Instance

Test Cases to fill-up.

6.6.5.7. LightweightM2M-1.1-int-735 -- Delete Object Instance

Test Cases to fill-up.

6.6.6. Firmware Update Object (ID 5) [750-799]

paragraph

6.6.6.1. LightweightM2M-1.1-int-751 -- Querying the readable resources

Test Case Id

LightweightM2M-1.1-int-751

Test Object

Client and Server

Test Case Description

Querying Information on the Firmware Update Object Instance for determining the current states and supported characteristics of the Client for such capability.

Preconditions

- The Client supports the Configuration C.8 as defined in Annex C
- The Client is registered with the Server
- The Fw Update Object Instance is in its initial state (State=0)

Test Procedure

A READ operation from the Server on the Firmware Update Object Instance requesting TLV format has been received by the client.

Normal flow:

4. READ (CoAP GET) operation is performed on the Firmware Update Object Instance

Pass-Criteria

1. In test step 1, the Server receives the status code "2.05" for READ operation success
2. In test step 1, the returned values regarding State (ID:3) and Update Result (ID:5) prove the Client FW update Capability is in initial state (State=Idle & Update Result= Initial Value).
3. In test step 1, the returned values regarding Firmware Update Protocol Support (ID:8) & Firmware Update Delivery Method (ID:9) allow to determine the supported characteristics of the Client FW Update Capability.

6.6.6.2. LightweightM2M-1.1-int-755 -- Setting the writable Resource Package

Test Case Id

LightweightM2M-1.1-int-755

Test Object

Client and Server

Test Case Description

Setting the writable Resource Package (ID:0) of Firmware Update Object Instance and verifying the triggered actions produce the expected results

Preconditions

- The Client supports the Configuration C.8 as defined in Appendix C
- The Client is registered with the Server
- According to Test LightweightM2M-1.1-int-751, the FW Update Object Instance supports the PUSH Delivery Method

Test Procedure

A WRITE operation from Server to set to NULL value ('\0') the Resource Package (ID:0) has been received by the client. The test verifies the FW Update Object Instance is reset to its Initial state. Then a WRITE operation is performed by the Server to load a valid image in the Client in setting the Package Resource.

The various Resources of this FW Update Objects MUST be set as expected by the Client.

Normal flow:

1. A WRITE (CoAP PUT) operation with a NULL value ('\0') is performed by the Server on the Package Resource (ID:0) of the FW Update Object Instance
2. The Server READs (CoAP GET) the FW Object Instance to get the values of the State (ID:3) and Update Result (ID:5) Resources
3. A WRITE (CoAP PUT) operation with a valid image is performed by the Server on the Package Resource (ID:0) of the FW Update Object Instance
4. The Server READs (CoAP GET) the FW Object Instance to get the values of the State (ID:3) and Update Result (ID:5) Resources

Pass-Criteria

1. In test step 1, the Server receives the success message "2.04" associated with the WRITE operation
2. In test step 2, the Server receives the success message "2.05" along with the value of State and Update Result Resources values.
3. In test step 2, the queried State and Update Result Resources values are both 0 (Idle / Initial value): FW Update Object Instance is in the Initial state.
4. In test step 3, the Server receives the success message "2.04" associated with the WRITE request for loading the firmware image.
5. In test step 4, the Server receives the success message "2.05" along with the State and Update Result Resources values.
6. In test step 4, the queried value of State resource is 2 (Downloaded) and the value of Update Result value is still 0 (Initial Value)

6.6.6.3. LightweightM2M-1.1-int-756 -- Setting the writable Resource Package URI

Test Case Id

LightweightM2M-1.1-int-756

Test Object

Client and Server

Test Case Description

Setting the writable Resource Package URI (ID:1) of Firmware Update Object Instance and verifying the triggered actions

produce the expected results

Preconditions

- The Client supports the Configuration C.8 as defined in Appendix C
- The Client is registered with the Server
- According to Test LightweightM2M-1.1-int-751, the FW Update Object Instance supports the PULL Delivery Method

Test Procedure

A WRITE operation from Server to set the empty string to the Resource Package URI (ID:0) has been received by the client. The test verifies the FW Update Object Instance is reset to its Initial state. Then a WRITE operation is performed by the Server to load a valid URI in the Client in setting the Package URI Resource.

The various Resources of this FW Update Objects MUST be set as expected by the Client.

Normal flow:

1. A WRITE (CoAP PUT) operation with an empty string value is performed by the Server on the Package Resource (ID:0) of the FW Update Object Instance
2. The Server READs (CoAP GET) the FW Object Instance to get the values of the State (ID:3) and Update Result (ID:5) Resources
3. A WRITE (CoAP PUT) operation with a valid image is performed by the Server on the Package Resource (ID:0) of the FW Update Object Instance
4. The Server READs (CoAP GET) the FW Object Instance to get the values of the State (ID:3) and Update Result (ID:5) Resources

Pass-Criteria

1. In test step 1, the Server receives the success message "2.04" associated with the WRITE operation
2. In test step 2, the Server receives the success message "2.05" along with the value of State and Update Result Resources values.
3. In test step 2, the queried State and Update Result Resources values are both 0 (Idle / Initial value): FW Update Object Instance is in the Initial state.
4. In test step 3, the Server receives the success message "2.04" associated with the WRITE request for the loaded image.
5. In test step 4, the Server receives the success message "2.05" along with the State and Update Result Resources values.
6. In test step 4, the queried value of State resource is 2 (Downloaded) and the value of Update Result value is still 0 (Initial Value)

6.6.6.4. LightweightM2M-1.1-int-760 --Basic Observation and notification on Firmware Update Object Resources

Test Case Id

LightweightM2M-1.1-int-760

Test Object

Client and Server

Test Case Description

Sending the observation policy to the Client for State Resource of Firmware Update Object (ID:5)

Preconditions

- The Client supports the Configuration C.8 as defined in Appendix C
- Device is switched on and operational
- Client is registered at the LwM2M server
- According to Test LightweightM2M-1.1-int-751, the FW Update Object Instance supports the PUSH Delivery Method
- Test LightweightM2M-1.1-int-755 already passed (an firmware image can be successfully loaded)
- The Client is in Initial State regarding the FW Update functionality (State=Idle)

Test Procedure

The Server establishes an Observation relationship with the Client to acquire notifications about State Resource of the Fw Update Object Instance

Normal flow:

1. The Server communicates to the Client pmin=2 and pmax=10 periods with a WRITE-ATTRIBUTE (CoAP PUT) operation at the FW Update Object Instance level.
2. The Server Sends OBSERVE (CoAP Observe Option) message to activate reporting on the State Resource (/5/0/3) of the FW Update Object Instance.
3. The Server delivers the firmware to the Client through a WRITE (CoAP PUT) operation on the Package Resource (/5/0/0)
4. The Client reports requested information with a NOTIFY message (CoAP response)

Pass-Criteria

1. In test step 1, the Server receives the success message "2.04" associated with the WRITE-ATTRIBUTE operation.
2. In test step 2, the Server receives the success message "2.05" associated with the OBSERVE operation, along with the value of State =Idle
3. In test step 3, the Server receives the success message "2.04" associated with the WRITE operation delivering the firmaware image.
4. In test step 4, the State Resource value returned by the Client in NOTIFY message is set to "Downloaded"

6.6.6.5. LightweightM2M-1.1-int-770 -- Successful Firmware update (via COAP)

Test Case Id

LightweightM2M-1.1-int-770

Test Object

Client and Server

Test Case Description

Perform a device firmware update remotely triggered by the LWM2M server in using Package Resource (CoAP mechanisms)

Preconditions

- The Client supports the Configuration C.8 as defined in Annex C
- According to Test LightweightM2M-1.1-int-751, the FW Update Object Instance supports the PULL Delivery Method
- The Client already passed the LightweightM2M-1.1-int-755 test
- Device is switched on and operational
- The Client is registered with the LWM2M server
- Firmware Update is available on the Server
- State is "0" (Idle) or "1" (Downloaded)

Test Procedure

After setting the Client in the Initial stage for supporting Firmware Update process, the Server delivers the Package to the Client before to trigger the installation of the downloaded firmware. Finally, the Server checks the result of the full process.

Normal flow:

1. Step 1 -- Package Delivery

a. The Server places the Client in the initial state of the FW Update process : A WRITE (CoAP PUT) operation with a NULL value ('\\0') is performed by the Server on the Package Resource (ID:0) of the FW Update Object Instance

b. The Server delivers the firmware to the Client through a WRITE (CoAP PUT) operation on the Package Resource (/5/0/0)

c. Polling (READ command) or Notification on Update Result and State Resources is performed, up to the time State Resource takes the 'Downloaded' value (2)

2. Step 2 -- Firmware Update

a. When the download is completed (State Resource value is '2' Downloaded) , the Server initiates a firmware update by triggering EXECUTE command on Update Resource (CoAP POST /5/0/2)

b. Polling (READ command) or Notification on Update Result and State Resources is performed, up to the time State Resource is turned back to Idle value (0) or Update Result Resource contains another value than the Initial one (0)

3. Step 3 -- Process verification

a. The Server READs Update Result ("/5/0/5") and State ("/5/0/3") Resources to know the result of the firmware update procedure.

b. The Server READs the Resource "Firmware Update" from the Object Device Instance ("/3/0/3")

Pass-Criteria

1. *Step 1 -- Package Delivery**

   a. In the test step 1.a, the Server receives the status code "2.04" for the WRITE success setting the Client in the FW update initial state.

   b. In the test step 1.b, The Server receives success message with either a "2.31" status code (Continue) or a final "2.04" status code.

   c. In the test step 1.c State Resource can take the value "1" (Downloading) during this sub-step and will take the value "2" at the Send (Downloaded)

   d. Update Result is "0" (Initial Value) during the whole step

2. *Step 2 -- Firmware Update**

   a. In test step 2.a, the Server receives a success message "2.04" (Changed) in response to the EXECUTE command

   b. In test step 2.b, the Server receives success message(s) "2.05" Contents along with a State Resource value of "3" (Updating) or "0" (Idle) and an Update Ressource value of "0" (Initial Value) or "1" (Firmware updated successfully).

3. *Step 3 -- Process verification**

   a. In test step 3.a, the Server receives success message(s) "2.05" Content" along with a State Resource value of "0" (Idle) and an Update Ressource value of "1" (Firmware updated successfully)

   b. In test step 3.b, the Server receives success message "2.05" Content" along with the expected value of the Resource Firmware Version from the Object Device Instance

6.6.6.6. LightweightM2M-1.1-int-771 -- Successful Firmware update (via alternative mechanism)

Test Case Id

LightweightM2M-1.1-int-771

Test Object

Client and Server

Test Case Description

Perform a device firmware update remotely triggered by the LwM2M server in using Package URI Resource

Preconditions

- The Client supports the Configuration C.8 as defined in Annex C

- According to Test LightweightM2M-1.1-int-751, the FW Update Object Instance supports the PUSH Delivery Method
- The Client passed the LightweightM2M-1.1-int-755 test
- Device is switched on and operational
- The Client is registered with the LwM2M server
- Firmware Update is available on the Server
- State is "0" (Idle) or "1" (Downloaded)

Test Procedure

After setting the Client in the Initial stage for supporting Firmware Update process, the Server delivers the Package URI to the Client, waits for the firmware to be downloaded and then triggers the installation of the downloaded firmware. Finally, the Server checks the result of the full process. Normal flow:

1. Step 1 -- Package Delivery

a. The Server places the Client in the initial state of the FW Update process : A WRITE (CoAP PUT) operation with an empty string value is performed by the Server on the Package URI Resource (ID:1) of the FW Update Object Instance

b. The Server delivers the Package URI to the Client through a WRITE (CoAP PUT) operation on the Package URI Resource (/5/0/1)

c. The Client downloads the firmware from the provided URI via an alternative mechanism (not CoAP)

d. Polling (successive READ commands) or Notification on Update Result and State Resources is performed, up to the time State Resource takes the 'Downloaded' value (2)

1. Step 2 -- Firmware Update

a. When the download is completed (State Resource value is '2' Downloaded) , the Server initiates a firmware update by triggering EXECUTE command on Update Resource (CoAP POST /5/0/2)

b. Polling (READ command) or Notification on Update Result and State Resources is performed, up to the time State Resource is turned back to Idle value (0) or Update Result Resource contains another value than the Initial one (0)

1. Step 3 -- Process verification

a. The Server READs Update Result ("/5/0/5") and State ("/5/0/3") Resources to know the result of the firmware update procedure.

b. The Server READs the Resource "Firmware Update" from the Object Device Instance ("/3/0/3")

Pass-Criteria

1. Step 1 -- Package Delivery

   a. In the test step 1.a, the Server receives the status code "2.04" for the WRITE success setting the Client in the FW update initial state.

   b. In the test step 1.b, the Server receives the status code "2.04" for the WRITE success setting the Package URI Client in the FW update Object Instance

   c. In the test step 1.c, The Server receives success message with either a "2.31" status code (Continue) or a final "2.04" status code.

d. In the test step 1.d State Resource can take the value "1" (Downloading) during this sub-step and will take the value "2" at the Send (Downloaded)

e. Update Result is "0" (Initial Value) during the whole test step 1

2. Step 2 -- Firmware Update

a. In test step 2.a, the Server receives a success message "2.04" (Changed) in response to the EXECUTE command

b. In test step 2.b, the Server receives success message(s) "2.05" Contents along with a State Resource value of "3" (Updating) or "0" (Idle) and an Update Ressource value of "0" (Initial Value) or "1" (Firmware updated successfully)

3. Step 3 -- Process verification

a. In test step 3.a, the Server receives success message(s) "2.05" Content" along with a State Resource value of "0" (Idle) and an Update Ressource value of "1" (Firmware updated successfully)

b. In test step 3.b, the Server receives success message "2.05" Content" along with the expected value of the Resource Firmware Version from the Object Device Instance

6.6.6.7. LightweightM2M-1.1-int-772 -- Error Case 1: firmware installation without downloaded package

Test Case Id

LightweightM2M-1.1-int-772

Test Object

Client and Server

Test Case Description

Try to perform a Device firmware installation when there is no downloaded firmware package

Preconditions

- The Client supports the Configuration C.8 as defined in Annex C
- Device is switched on and operational
- Client is registered with the LwM2M server
- Firmware Update is available on the Server
- The State Resource (/5/0/3) of the FW Object Instance is different from "2" (Downloaded)

Test Procedure

The Server initiates a firmware installation on the Client while this Client has no downloaded package in it.

Normal flow:

1. The Server sends a READ operation (CoAP GET /5/0) to the Client on the FW Update Object Instance to obtain the

values of the State and Update Resources.

2. The Client receives an EXECUTE operation on the Update Resource (CoAP POST /5/0/2) of the FW Update Object Instance.

3. The Server sends a READ operation again (CoAP GET /5/0/3) to the Client on the FW Update Object Instance to obtain the State and the Update Resource values

Pass-Criteria

1. In test step 1, the Server receives a success message ("2.05" Content) associated to its READ command along with a State Resource value which is not "2" (Downloaded) and a valid (0 to 9) Update Resource value

2. In test step 2, the Server receives the status code "4.05" for method not allowed associated to its EXECUTE command.

3. In test 3, the Server receives a success message ("2.05" Content associated to is READ command along with a State Resource value and an Update Result Resource value, identical to the ones retrieved in Pass-Criteria A. The firmware has not been installed.

6.6.6.8. LightweightM2M-1.1-int-773 -- Error Case 2: shortage of storage memory

Test Case Id

LightweightM2M-1.1-int-773

Test Object

Client and Server

Test Case Description

Try to perform a Device firmware update remotely triggered by the LWM2M Server with a firmware package exceeding the Client storage memory capacity.

Preconditions

- The Client supports the Configuration C.8 as defined in Annex C
- Prepare a package with a size exceeding the Client storage memory capacity
- Device is switched on and operational
- Client is registered with the LwM2M server
- Firmware Update is available on the Server
- The State Resource (/5/0/3) of the FW Object Instance is set to "0" (Idle)

Test Procedure

A Package delivery initiated by a Server in using either PULL or PUSH delivery methods with a firmware package exceeding the Client storage memory capacity, is received by the Client

Normal flow:

1. The Server verifies through a READ (CoAP GET) command on /5/0/3 (State) the FW Update Object Instance of the Client is in Idle State
2. The Server delivers the firmware package to the Client either through a WRITE (CoAP PUT) operation in the Package Resource (/5/0/0) or through a WRITE operation of an URI in the Package URI Resource.
3. The firmware downloading process is runing The Server sends repeated READs or OBSERVE on State and Update Result Resources (CoAP GET /5/0) of the FW Update Object Instance to determine when the download is completed or if an error occured.Before the Send of download, the device runs out of storage and cannot finish the download
4. When the Package delivery is stopped the server READs Update Result to know the result of the firmware update procedure.

Pass-Criteria

1. In test step 1, the Server receives the status code "2.05 " (Content) for the READ success command, along with the State Resource value of "0" (Idle)
2. In test step 2., the Server receives the status code "2.04" (Changed) for the WRITE command setting either the Package URI Resource or setting the Package Resource, according to the chosen firmware delivery method.
3. In test step 3., the State Resource retrieved with a value of "1" from successive Server READs or Client NOTIFY messages, indicates the download stage of the Package Delivery is engaged
4. In test step 3., the Update Result Resource (/5/0/5) retrieved from successive Server READs or Client NOTIFY messages will take the value "2" indicating an error occurred during the downloading process related to shortage of storage memory The State Resource value never reaches the Downloaded value ("3")
5. In test step 4., the success READ message(s) (status code "2.05" Content) on State Resource with value "0" (Idle) and Update Result Resource with value "2" indicates the firmware Package Delivery aborted due to shortage of storage memory.

6.6.6.9. LightweightM2M-1.1-int-774 -- Error Case 3: out of memory

Test Case Id

LightweightM2M-1.1-int-774

Test Object

Client and Server

Test Case Description

Try to perform a Device firmware update remotely triggered by the LWM2M Server while the Client runs out of RAM during download

Preconditions

- The Client supports the Configuration C.8 as defined in Annex C
- Prepare a firmware package to make the Client running out of RAM when downloading it.
- Device is switched on and operational

- Client is registered at the LwM2M server
- Firmware Update is available on the Server
- The State Resource (/5/0/3) of the FW Object Instance is set to "0" (Idle)

Test Procedure

A Package delivery initiated by a Server using either PULL or PUSH delivery methods is received by the Client. This firmware package will make the Client to run out of RAM during its download,

Normal flow:

1. The Server verifies through a READ (CoAP GET) command on /5/0/3 (State) the FW Update Object Instance of the Client is in Idle State
2. The Server delivers the firmware package to the Client either through a WRITE (CoAP PUT) operation in the Package Resource (/5/0/0) or through a WRITE operation of an URI in the Package URI Resource.
3. The firmware download process is runing The Server sends repeated READs or OBSERVE on State and Update Result Resources (CoAP GET /5/0) of the FW Update Object Instance to determine when the download is completed or if an error occured.Before the Send of download, the Client runs out of RAM and cannot finish the download
4. When the Package delivery is stopped the server READs Update Result to know the result of the firmware update procedure.

Pass-Criteria

1. In test step 1, the Server receives the status code "2.05 " (Content) for the READ success command, along with the State Resource value of "0" (Idle)
2. In test step 2., the Server receives the status code "2.04" (Changed) for the WRITE command setting either the Package URI Resource or setting the Package Resource, according to the chosen firmware delivery method.
3. In test step 3., the State Resource retrieved with a value of "1" from successive Server READs or Client NOTIFY messages, indicates the download stage of the Package Delivery is engaged
4. In test step 3., the Update Result Resource (/5/0/5) retrieved from successive Server READs or Client NOTIFY messages will take the value "2" indicating an error occurred during the download process related to shortage of RAM The State Resource value never reaches the Downloaded value ("3")
5. In test step 4., the success READ message(s) (status code "2.05" Content) on State Resource with value "0" (Idle) and Update Result Resource with value "2" indicates the firmware Package Delivery aborted due to shortage of RAM.

6.6.6.10. LightweightM2M-1.1-int-775 -- Error Case 4: Connection lost during download (package URI)

Test Case Id

LightweightM2M-1.1-int-775

Test Object

Client and Server

Test Case Description

Try to perform a Device firmware update remotely triggered by the LWM2M server while connection is lost during download

Preconditions

- The Client supports the Configuration C.8 as defined in Annex C
- Prepare a method for loosing the connection when firmware package download is engaged.
- Device is switched on and operational
- Client is registered at the LwM2M server
- Firmware Update is available on the Server
- The State Resource (/5/0/3) of the FW Object Instance is set to "0" (Idle)

Test Procedure

A Package delivery initiated by a Server in using either the PULL delivery methods is received by the Client

Normal flow:

1. The Server verifies through a READ (CoAP GET) command on /5/0/3 (State) the FW Update Object Instance of the Client is in Idle State
2. The Server delivers the firmware package to the Client through a WRITE operation of an URI in the Package URI Resource.
3. The Server sends repeated READs or OBSERVE on State and Update Result Resources (CoAP GET /5/0) of the FW Update Object Instance to determine when the download is completed or if an error occured.Before the Send of download, the connection is intentionnaly lost and the download cannot be finished.
4. When the Package delivery is stopped the Server READs Update Result to know the result of the firmware update procedure.

Pass-Criteria

1. In test step 1., the Server receives the status code "2.05 " (Content) for the READ success command, along with the State Resource value of "0" (Idle)
2. In test step 2., the Server receives the status code "2.04" (Changed) for the WRITE command setting the Package URI Resource according to the PULL firmware delivery method.
3. In test step 3., the State Resource value set to "1" retrieved from successive Server READs or Client NOTIFY messages, indicates the Package Delivery process is engaged in a Download stage
4. In test step 3., the Update Result Resource (/5/0/5) retrieved from successive Server READs or Client NOTIFY messages will take the value "4" indicating an error occurred during the downloading process related to connection lost
5. In test step 4., the success READ message(s) (status code "2.05" Content) on State Resource with value "0" (Idle)

and Update Result Resource with value "4" indicates the firmware Package Delivery aborted due to connection lost dur the Package delivery.

6.6.6.11. LightweightM2M-1.1-int-776 -- Error Case 5: Package Integrity check failure

Test Case Id

LightweightM2M-1.1-int-776

Test Object

Client and Server

Test Case Description

Try to perform a device firmware update remotely triggered by the LWM2M Server while the downloaded package does not pass the integrity check

Preconditions

- The Client supports the Configuration C.8 as defined in Annex C
- Prepare a firmware package with a wrong Data Integrity Check in it.
- Device is switched on and operational
- Client is registered at the LwM2M server
- Firmware Update is available on the Server
- The State Resource (/5/0/3) of the FW Object Instance is set to "0" (Idle)

Test Procedure

A Package delivery initiated by a Server in using either PULL or PUSH delivery methods is received by the Client. This firmware package has been generated with a wrong Data Integrity Check,

Normal flow:

1. The Server verifies through a READ (CoAP GET) command on /5/0/3 (State) the FW Update Object Instance of the Client is in Idle State
2. The Server delivers the firmware package to the Client either through a WRITE (CoAP PUT) operation in the Package Resource (/5/0/0) or through a WRITE operation of an URI in the Package URI Resource.
3. The Server sends repeated READs or OBSERVE on State and Update Result Resources (CoAP GET /5/0) of the FW Update Object Instance to determine when the download is completed or if an error occured. The firmware package Integry Check failure stopped the download process.
4. When the Package delivery is stopped the server READs Update Result to know the result of the firmware update procedure.

Pass-Criteria

1. In test step 1, the Server receives the status code "2.05 " (Content) for the READ success command, along with the

State Resource value of "0" (Idle)

2. In test step 2., the Server receives the status code "2.04" (Changed) for the WRITE command setting either the Package URI Resource or setting the Package Resource, according to the chosen firmware delivery method.
3. In test step 3., the State Resource value set to "1" retrieved from successive Server READs or Client NOTIFY messages, indicates the Package Delivery process is maintained in Downloading stage
4. In test step 3., the Update Result Resource (/5/0/5) retrieved from successive Server READs or Client NOTIFY messages will take the value "5" indicating an error occurred during the downloading process related to the failure of the firmware package integrity check
5. In test step 4., the success READ message(s) (status code "2.05" Content) on State Resource with value "0" (Idle) and Update Result Resource with value "5" indicates the firmware Package Delivery aborted due to a Firmware Package Integrity failure.

6.6.6.12. LightweightM2M-1.1-int-777 -- Error Case 6: unsupported package type

Test Case Id

LightweightM2M-1.1-int-777

Test Object

Client and Server

Test Case Description

Try to perform a device firmware update remotely triggered by the LWM2M Server with an unsupported package type

Preconditions

- The Client supports the Configuration C.8 as defined in Annex C
- Prepare a firmware package containing a package type not supported by the Client
- Device is switched on and operational
- Client is registered at the LwM2M server
- Firmware Update is available on the Server
- The State Resource (/5/0/3) of the FW Object Instance is set to "0" (Idle)

Test Procedure

A Package delivery initiated by a Server in using either PULL or PUSH delivery methods is received by the Client. The package type is not supported by the Client.

Normal flow:

1. The Server verifies through a READ (CoAP GET) command on /5/0/3 (State) the FW Update Object Instance of the Client is in Idle State
2. The Server delivers the firmware package to the Client either through a WRITE (CoAP PUT) operation in the

Package Resource (/5/0/0) or through a WRITE operation of an URI in the Package URI Resource.

3. The Server sends repeated READs or OBSERVE on State and Update Result Resources (CoAP GET /5/0) of the FW Update Object Instance to determine when the download is completed or if an error occured. The Download cannot be finished since the firmware package type is not supported by the Client
4. When the Package delivery is stopped the server READs Update Result to know the result of the firmware update procedure.

Pass-Criteria

1. In test step 1, the Server receives the status code "2.05 " (Content) for the READ success command, along with the State Resource value of "0" (Idle)
2. In test step 2., the Server receives the status code "2.04" (Changed) for the WRITE command setting either the Package URI Resource or setting the Package Resource, according to the chosen firmware delivery method.
3. In test step 3., the State Resource value set to "1" retrieved from successive Server READs or Client NOTIFY messages, indicates the Package Delivery process is in Downloading stage
4. In test step 3., the Update Result Resource (/5/0/5) retrieved from successive Server READs or Client NOTIFY messages will take the value "6" indicating an error occurred during the downloading process related to the firmware package type not supported by the Client.
5. In test step 4., the success READ message(s) (status code "2.05" Content) on State Resource with value "1" (Downloading) and Update Result Resource with value "6 indicates the firmware Package Delivery aborted due to a firmware package type not supported by the Client.

6.6.6.13. LightweightM2M-1.1-int-778 -- Error Case 7: invalid URI (package URI)

Test Case Id

LightweightM2M-1.1-int-778

Test Object

Client and Server

Test Case Description

Try to perform a device firmware update remotely triggered by the LWM2M server with an invalid URI

Preconditions

- The Client supports the Configuration C.8 as defined in Annex C
- According to Test LightweightM2M-1.0-int-751, the FW Update Object Instance supports the PULL Update Delivery Method
- Prepare an invalid URI from which the Client will try to fetch a firmware package.
- Device is switched on and operational
- Client is registered with the LwM2M server

- Firmware Update is available on the ServerThe State Resource (/5/0/3) of the FW Object Instance is set to "0" (Idle)

Test Procedure

A Package delivery initiated by a Server in using the PULL delivery methods is received by the Client. The URI provided is a fake one.

Normal flow:

1. The Server verifies through a READ (CoAP GET) command on /5/0/3 (State) the FW Update Object Instance of the Client is in Idle State
2. The Server initiates a firmware package delivery to the Client through a WRITE operation of an invalid URI in the Package URI Resource.
3. The Server sends repeated READs or OBSERVE on State and Update Result Resources (CoAP GET /5/0) of the FW Update Object Instance to determine when the download is completed or if an error occured. The download process is stopped by the Client due to the usage of a bad URI.
4. When the Package delivery is stopped the server READs Update Result to know the result of the firmware update procedure.

Pass-Criteria

1. In test step 1, the Server receives the status code "2.05 " (Content) for the READ success command, along with the State Resource value of "0" (Idle)
2. In test step 2., the Server receives the status code "2.04" (Changed) for the WRITE command setting the Package URI Resource
3. In test step 3., the State Resource value set to "1" retrieved from successive Server READs or Client NOTIFY messages, indicates the Package Delivery process is maintained in Downloading stage
4. In test step 3., the Update Result Resource (/5/0/5) retrieved from successive Server READs or Client NOTIFY messages will take the value "7" indicating an error occurred during the downloading process related to the usage of a bad URI
5. In test step 4., the success READ message(s) (status code "2.05" Content) on State Resource with value "0" (Idle) and Update Result Resource with value "7" indicates the firmware Package Delivery aborted due to the connection to an Invalid URI for the firmware package delivery.

6.6.6.14. LightweightM2M-1.1-int-779 -- Error Case 8: Unsuccessful Firmware Update

Test Case Id

LightweightM2M-1.1-int-779

Test Object

Client and Server

Test Case Description

Perform a device firmware update remotely triggered by the LWM2M server in using Package Resource (CoAP

mechanisms) or but with installation failure

Preconditions

- The Client supports the Configuration C.8 as defined in Annex C
- Prepare a firmware package which can be downloaded but cannot be successfully installed.
- Device is switched on and operational
- Client is registered with the LwM2M Server
- Firmware Update is available on the Server
- The State Resource (/5/0/3) of the FW Object Instance is set to "0" (Idle)

Test Procedure

A Package delivery initiated by a Server in using either PULL or PUSH delivery methods is received by the Client. But its installation will fail.

Normal flow:

1. Step 1 -- Package Delivery

a. The Server verifies through a READ (CoAP GET) command on /5/0/3 (State) the FW Update Object Instance of the Client is in Idle State

b. The Server retrieves (CoAP GET) the initial value of the Firmware Version Resource from the Object Device Instance for verification in the Pass Criteria (C)

c. The Server delivers the firmware package to the Client either through a WRITE (CoAP PUT) operation in the Package Resource (/5/0/0) or through a WRITE operation of an URI in the Package URI Resource.

d. Polling (successive READ commands) or Notification on Update Result and State Resources is performed, up to the time State Resource takes the 'Downloaded' value (2)

2. Step 2 -- Installation Failure

a. When the download is completed (State Resource value is '2' Downloaded) , the Server initiates a firmware update by triggering EXECUTE command on Update Resource (CoAP POST /5/0/2)

b. Polling (READ command) or Notification on Update Result and State Resources is performed, up to the time State Resource is turned back to 2 (Downloaded) or the Update Result Resource contains the value "8" (Firmware update failed)

3. Step 3 -- Process Verification

a. The server READs Update Result & State Resources to know the result of the firmware update procedure.

b. The Server READs the Firmware Version Resource from the Object Device Instance

Pass-Criteria

1. *Package Delivery**

   a. In test step 1.a, the Server receives the status code "2.05 " (Content) for the READ success command, along with the State Resource value of "0" (Idle)

   b. In test step 1.b, the Server receives the status code "2.05 " (Content) for the READ success command, along with the initial value of the Firmware version Resource available from the Object Device Instance.

c. In test step 1.c, the Server receives the status code "2.04" (Changed) for the WRITE command setting either the Package URI Resource or setting the Package Resource, according to the chosen firmware delivery method.

d. In at this Send of test step 1.d, the State Resource take the value "2" (Downloaded)

2. *Installation failure**

a. In test step 2.a, the Server receives a success message "2.04" (Changed) in response to the EXECUTE command

b. In test step 2.b, the Server receives success message(s) "2.05" Contents along with a State Resource value of "3" (Updating) or "2" (Downloaded) and an Update Ressource value of "0" (Initial Value) and "8" at the Send (Firmware updated failure)

3. *Process Verification**

a. In test step 3.a, the Server receives success message(s) "2.05" Content" along with a State Resource value of "2" (Downloaded) and an Update Ressource value of "8" (Firmware updated failed)

b. In test step 3.b the Server receives success message(s) "2.05" Content" along with a Firmware Version Resource value form the Object Device Instance which has not changed compared to the one retrieved in Pass Criteria A.b

6.6.6.15. LightweightM2M-1.1-int-780 -- Error Case 9: Unsupported protocol

paragraph

6.6.7. LwM2M Location Object (ID:6) [800-849]

6.6.7.1. LightweightM2M-1.1-int-801 -- Querying the readable resources of object

Test Case Id

LightweightM2M-1.1-int-801

Test Object

Client and Server

Test Case Description

Querying the readable resources of Object Location ID:6 Instance and verifying the received values are as expected and coherent.

Preconditions

- The Client supports the Configuration C.7 as defined in Annex C
- Client is registered with the Server

Test Procedure

A READ operation from Server on the Location Object Instance has been received by the Client.

Normal flow:

1. READ (CoAP GET) is performed by the Server on the Location Object Instance of the Client

Pass-Criteria

1. In test step 1., the Client receives a READ (CoAP GET) command from the Server on the Location Object Instance
2. In test step 1., the Server receives the status code "2.05" for READ message success
3. In test step 1., along with the success message, the mandatory Resources (Latitude, Longitude, Timestamp) and optional ones, are received by the Server with expected values in compliance with LwM2M technical specification 1.0

6.6.7.2. LightweightM2M-1.1-int-805 -- Setting the writable resources

There are no writable resources for this object.

6.6.7.3. LightweightM2M-1.1-int-810 -- Basic Observation and notification on Location Object Instance

Test Case Id

LightweightM2M-1.1-int-810

Test Object

Client and Server

Test Case Description

Sending the observation policy to the device for resources of Object Location ID:6

Preconditions

- The Client supports the Configuration C.7 as defined in Annex C
- Device is switched on and operational
- Client is registered with the LWM2M server

Test Procedure

The Server establishes an Observation relationship with the Client to acquire condition notifications about the Location Object Instance observable resources.

Normal flow:

1. The Server communicates to the Client pmin=2 pmax=10 period threshold values with a WRITE ATTRIBUTE (CoAP PUT) operation at the Location Object Instance level
2. The Server sends OBSERVE (CoAP Observe Option) message to activate reporting on the Location Object Instance.
3. The Client reports requested information with a NOTIFY message (CoAP response)

Pass-Criteria

1. In test step 1., the Server received a WRITE ATTRIBUTE command (CoAP PUT) targeting the Location Object Instance with the proper pmin=2 and pmin=10 parameters.
2. In test step 1., the Server received the success message (2.04 Changed) in response to the WRITE ATTRIBUTE command
3. In test step 2., the Client received the OBSERVE operation targeting the Location Object Instance
4. In test step 2., in response to its OBSERVE operation, the Server receives the success message (Content 2.05) along with the initial values of the Location Object Instance
5. In test step 3., based on pmin/pmax periods parameters received in test step 1., the Client reports information to the Server with NOTIFY messages containing the Location Object Instance updated values.
6. In test step 3., the values received by the Server along with the success message (Content 2.05) MUST be as expected : at less the Mandatory Timestamp Resource MUST have admissible values according to the pmin and pmax parameters.

6.6.7.4. LightweightM2M-1.1-int-811 -- Extended Observation and notification of Location Object Instance

Test Cases to fill-up.

6.6.7.5. LightweightM2M-1.1-int-820 -- Create Multiple Resource Instances

Test Cases to fill-up.

6.6.7.6. LightweightM2M-1.1-int-830 -- Create Object Instance

Test Cases to fill-up.

6.6.7.7. LightweightM2M-1.1-int-835 -- Delete Object Instance

Test Cases to fill-up.

6.6.8. Connectivity Statistics (ID 7) [900-949]

paragraph

6.6.8.1. LightweightM2M-1.1-int-901 -- Querying a Data Collection from Connectivity Object Instance

Test Case Id

LightweightM2M-1.1-int-901

Test Object

Client and Server

Test Case Description

Starting and stopping after few seconds a data collection, then querying the readable resources of Connectivity Statistics Object ID:7 Instance and verifying the received values are as expected and coherent

Preconditions

- The Client supports the Configuration C.9 as defined in Annex C
- Client is registered with the Server

Test Procedure

A READ operation from Server on the Connectivity Statistics Object Instance has been received by the Client after the data collection has been triggered then stopped after few seconds.

Normal flow:

1. EXECUTE operation (CoAP POST) is performed on the Start Resource of the Connectivity Statistics Object Instance
2. After few seconds, an EXECUTE operation (CoAP POST) is performed on the Stop Resource of the Connectivity Statistics Object Instance
3. A READ (CoAP GET) operation is performed by the Server on the Connectivity Object Instance of the Client

Pass-Criteria

1. In test step 1., the Client receives a Start command for the Connectivity Statistics Object Instance (CoAP POST)
2. In test step 1., the Server receives the success message ("2.4" Changed) in response of its EXECUTE command on the Client
3. In test step2., the Client receives a Stop command for the Connectivity Statistics Object Instance (CoAP POST)
4. In test step 2., the Server receives the success message ("2.4" Changed) in response of its EXECUTE command on the Client
5. In test step 3., the Client receives a READ operation on the Connectivity Object Instance
6. In test step 3., the Server receives the status code "2.05" for the READ message success
7. In test step 3., along with the success message, the Server receives the Connectivity Statistics Object Instance values according to the Client preferred data format.
8. In test step 3, the received Connectivity Statistics Object Instance contains expected values regarding the data collection represented by the Resources 0 to 5 of the Connectivity Statistics Object Instance

6.6.8.2. LightweightM2M-1.1-int-905 -- Setting the writable resources

Test Case Id

LightweightM2M-1.1-int-905

Test Object

Client and Server

Test Case Description

Query the readable Resources of the Connectivity Statistics Object ID:7 Instance, before setting the Collection Period to another value than the initial value one. Then query the readable Resources again and verify the Collection Period contains the expected value.

Preconditions

- The Client supports the Configuration C.9 as defined in Annex C
- Client is registered with the Server

Test Procedure

A WRITE operation is received by the Client for setting the Collection Period Resource of the Connectivity Statistics Object Instance. This WRITE operation is framed by 2 READ operations on the same Object Instance.

Normal flow:

1. A READ (CoAP GET) operation is performed by the Server on the Connectivity Object Instance of the Client
2. A WRITE (CoAP PUT) operation is performed on the Client targeting the Collection Period Resource (ID:8) of the Connectivity Object Instance with a value different of the one collected in step 2.
3. A new READ (CoAP GET) operation is performed by the Server on the Connectivity Object Instance of the Client

Pass-Criteria

1. In test step 1., the Client receives a READ operation on the Connectivity Object Instance
2. In test step 1., the Server receives the status code "2.05" for the READ success message along with the Connectivity Statistics Object Instance value according to the Client preferred data format.
3. In step 2, the Client receives a WRITE command (CoAP PUT) targeting the Collection Period of the Connectivity Statistics Object Instance with a value different of the one collected in pass B.
4. In test step 2, the Server received receives a success message (2.04 Changed) related to its WRITE operation
5. In test step 3., the Client receives a new READ operation on the Connectivity Object Instance
6. In step 3, the Server receives success message (2.05 Content) and the requested value in the LwM2M Client preferred data format
7. The value of the Collection Period Resource value collected in Pass F, correspond to the value which was set in step 2

6.6.8.3. LightweightM2M-1.1-int-910 -- Basic Observation and notification on Connectivity Monitoring Object Instance

Test Case Id

LightweightM2M-1.1-int-910

Test Object

Client and Server

Test Case Description

Sending the observation policy to the device for resources of Connectivity Object ID:7 Instance

Preconditions

- The Client supports the Configuration C.9 as defined in Annex C
- Device is switched on and operational
- Client is registered with the LwM2M server

Test Procedure

The Server establishes an Observation relationship with the Client to acquire condition notifications about the Connectivity Statistics Instance.

Normal flow:

1. The Server communicates to the Client pmin=2 & pmax=10 periods with a WRITE ATTRIBUTE (CoAP PUT) operation at the Connectivity Statistics Instance level.
2. The Server set (CoAP PUT) the Collection Period Resource to 0 in the Connectivity Statistics Object Instance.
3. The Server sends OBSERVE (CoAP Observe Option) message to activate reporting on the Connectivity Statistics Instance.
4. The Server starts the data collection on the Connectivity Staistics Object Instance by triggering the Start Resource of this Object Instance (CoAP POST)
5. The Client reports requested information with NOTIFY messages (CoAP response)
6. Several NOTIFY messages are received by the Server related to the Connectivity Statistics Object Instance
7. The Server stops the data collection on the Connectivity Staistics Object Instance by triggering the Stop Resource of this Object Instance (CoAP POST)
8. The Server stops the OBSERVATION process

Pass-Criteria

1. In test step 1., the Server received a WRITE ATTRIBUTE command (CoAP PUT) targeting the Connectivity Statistics Object Instance with the proper pmin=2 and pmin=10 parameters.
2. In test step 1., the Server received the success message (2.04 Changed) in response to the WRITE ATTRIBUTE command
3. In test step 2 the Client received the OBSERVE operation targeting the Connectivity Statistics Object Instance

4. In test step2, in response to its OBSERVE operation, the Server receives the success message (Content 2.05) along with the initial values of the Connectivity Statistics Object Instance
5. In test step 3, based on pmin/pmax periods parameters received in test step 1., the Client reports information to the Server with NOTIFY messages containing the updated values of the Connectivity Statistics.
6. In test step 3., the values returned by the Server along with the success message (Content 2.05) MUST be as expected.

6.6.8.4. LightweightM2M-1.1-int-911 -- Extended Observation and notification of Connectivity Statistics Object Instance

Test Cases to fill-up.

6.6.8.5. LightweightM2M-1.1-int-920 -- Create Multiple Resource Instances

Test Cases to fill-up.

6.6.8.6. LightweightM2M-1.1-int-930 -- Create Object Instance

Test Cases to fill-up.

6.6.8.7. LightweightM2M-1.1-int-935 -- Delete Object Instance

Test Cases to fill-up.

6.6.9. Multi-Servers Context [950-999]

6.6.9.1. LightweightM2M-1.1-int-950 -- Multi-Servers Registration

Test Case Id

LightweightM2M-1.1-int-950

Test Object

Client and Server

Test Case Description

Test the Client capability to register 2 and connect 2 different Servers

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.14 as defined in Annex and containing 2 (Registred-)Server accounts
- Device is turned-on
- The Boostrap Information is available in the Client with 2 Server-Accounts to which the Client is likely to register, and support of the Access Control Object)

Test Procedure

The Client automatically registers with 2 Servers in using the 2 Server Accounts available in the bootstrapped configuration

Normal flow:

1. Registration message (CoAP POST) is sent from the Client to Server #1
2. Registration message (CoAP POST) is sent from the Client to Server #2

Pass-Criteria

1. In test step 1., the Server receives the REGISTER command along with the information related to the Server Account #1 (see LwM2M-1.0-int-101 Test Case for more)
2. In test step 1., the Client receives the "Success" message from Server #1 (2.01 Created)
3. In test step 2., the Server receives the REGISTER command along with the information related to the Server Account #2 (see LwM2M-1.0-int-101 Test Case for more)
4. In test step 2., the Client receives the "Success" message from Server #2 (2.01 Created)

6.6.9.2. LightweightM2M-1.1-int-951 -- Multi-Servers & Attributes

Test Case Id

LightweightM2M-1.1-int-951

Test Object

Client and Server

Test Case Description

Test the Client capability to identify and to report information according to a specific Server context

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.14 as defined in Annex and containing 2 (Registered)Server accounts as well as simple Access Control Structure
- Device is turned-on
- The Bootstrap Information is available in the Client with 2 Server-Accounts to which the Client is likely to register
- The Client pass the test case LightweightM2M-1.1-int-950 and is connected to Server#1 et Server#2

Test Procedure

Server #1 & Server#2 set different Observation period values for the Device Object Instance; A check is performed to verify the effective setting

Normal flow:

1. The Server #1 communicates to the Client pmin=2 and pmax=10 periods with a WRITE-ATTRIBUTE (CoAP PUT) operation at the Device Object Instance level.
2. The Server #2 communicates to the Client pmin=15 and pmax=50 periods with a WRITE-ATTRIBUTE (CoAP PUT) operation at the Device Object Instance level.
3. The Server #1 sends a DISCOVER command to the Client
4. The Server #2 send a DISCOVER command to the Client

Pass-Criteria

1. In test step 1., the Server#1 receives the "Success" message from the Client (2.04 Changed)
2. In test step 2., the Server#2 receives the "Success" message from the Client (2.04 Changed)
3. In test step 3., the Server#1 receives the "Success" message from Client (2.05 Content) related to its DISCOVER command along with the payload containing the following information regarding the Device Object Instance :

</3/0>;pmin=2;pmax=10,</3/0/0>,</3/0/1>,</3/0/2>,</3/0/3>,</3/0/11>,</3/0/16>

4. In test step 4., the Server#2 receives the "Success" message from Client (2.05 Content) related to the DISCOVER command along with the payload containing the following information regarding the Device Object Instance :

</3/0>;pmin=15;pmax=50,</3/0/0>,</3/0/1>,</3/0/2>,</3/0/3>,</3/0/11>,</3/0/16>

6.7. LwM2M Additional Objects Test cases [1000-2099]

6.7.1. Lock and Wipe Object (ID 8) [1000-1099]

paragraph

6.7.2. Software Management Object (ID 9) [1100-1199]

paragraph

6.7.3. Connectivity Management Objects (ID 10, 11, 12, 13) [1200-1499]

6.7.3.1. Cellular Network Connectivity ID:10 [1200-1249]

paragraph

6.7.3.1.1. LightweightM2M-1.1-int-1200 -- Querying the readable resources of Object ID:10

Test Case Id

LightweightM2M-1.1-int-1200

Test Object

Client and Server

Test Case Description

Querying the readable resources of Cellular network connectivity object ID:10 and verifying the received values are as expected and coherent regarding the Configuration C.10

Preconditions

- The Client supports the Configuration C.10 as defined in Annex C
- The Client is registered with the Server

Test Procedure

A READ operation from Server on the Cellular Network Connectivity Object Instance requesting TLV format is sent to the Client.

Normal flow:

1. READ (CoAP GET /10/0) operation on the Instance of the Object ID:10 is received by the Client and its answer is sent back to the Server

Pass-Criteria

1. In test step 1 :the Server receives the success message (2.05 Content) associated to its READ request
2. In step 1: the values returned by the Client in TLV format are consistent with the Configuration C 10 used for that test (PSM timer, Active Timer, eDRX parameters for WB-S1 mode, eDRX parameters for NB-S1 mode and Serving PLMN rate Control…).

6.7.3.1.2. LightweightM2M-1.1-int-1201 -- Querying the readable resources of Object ID:10 in version 1.1

Test Case Id

LightweightM2M-1.1-int-1201

Test Object

Client and Server

Test Case Description

Querying the readable resources of Cellular network connectivity object ID:10 version 1.1 and verifying the received values are as expected and coherent regarding the Configuration C.11

Preconditions

- The Client supports the Configuration C.11 as defined in Annex C
- The Client is registered with the Server in indicating the Object ID:10 is in version 1.1

Test Procedure

A DISCOVER operation is requested by the Server to verify the Object ID:10 is in version 1.1.

A READ operation from Server on the Cellular Network Connectivity Object Instance requesting TLV format is sent to the Client.

Normal flow:

1. DISCOVER (CoAP GET Accept:40) operation is performed by the Server on Object ID:10
2. READ (CoAP GET /10/0) operation on the Instance of the Object ID:10 is received by the Client and its answer is sent back to the Server

Pass-Criteria

1. In test step 1, the Server -- along with the success message 2.05 -- received the information related to Object ID:10 including

</10>;ver="1.1", </10/0>, </10/0/4>, </10/0/5>, </10/0/6>, </10/0/8>, </10/0/9>, </10/0/11>, </10/0/13>, </10/0/14>

2. In test step 2 :the Server receives the success message (2.05 Content) associated to its READ request on Object Instance /10/0
3. In test step 2: the values returned by the Client in TLV format are consistent with the Configuration C 10. used for that test (Activated Profile Names, PSM timer, Active Timer, eDRX parameters for WB-S1 mode, eDRX parameters for NB-S1 mode, Serving PLMN rate Control, Power Saving Modes, Active Power Saving Modes).

6.7.3.1.3. LightweightM2M-1.1-int-1202 -- Setting the Power Saving Mode Resources of Object ID:10 (version 1.1 only)

Test Case Id

LightweightM2M-1.1-int-1202

Test Object

Client and Server

Test Case Description

Setting the writable Resources of Cellular network connectivity Object ID:10 in version 1.1

Preconditions

- The Client supports the Configuration C.11 as defined in Annex C
- The Client is registered with the Server in indicating the Object ID:10 is in version 1.1
- The Power Saving Modes of Device is not 0

Test Procedure

A DISCOVER operation is requested by the Server to verify the Object ID:10 is in version 1.1.

A WRITE operation from Server on the Cellular Network Connectivity Object Instance requesting TLV format is sent to the Client to set the power saving mode. This test has to be run for the following resources:

a. Active Power Saving Modes

Normal flow:

1. DISCOVER (CoAP GET Accept:40) operation is performed by the Server on Object ID:10
2. A READ (CoAP GET /10/0/13) operation on the Instance 0 of the Object ID:10 is performed by the Server
3. A WRITE operation is performed by the Server on the "Active Power Saving Modes" Resource (CoAP PUT/POST 10/0/14) of the Object ID:10 Instance 0, in using the set of values contains in the 1210-SetOfValues sample below.The data format TLV is used. (11542)
4. A READ (CoAP GET /10/0/14) operation is performed by the Server on the Instance 0 of the Object ID:10

Pass-Criteria

1. In test step 1, the Server -- along with the success message 2.05 -- received the information related to Object ID:10 including

</10>;ver="1.1", </10/0>,</10/0/6>,</10/0/11>,</10/0/13>,</10/0/14>

2. In test step 2 : the Server receives the success message (2.05 Content) and the requested value of the "Power Saving Modes" Resource (/10/0/13). This value MUST mach the value present in the C.11 configuration.
3. In step 3, the Server receives the success message (2.04 Changed) related to the WRITE command
4. In test step 4 : the Server receives the success message (2.05 Content) associated to its READ request and the received value for the "Active Power Saving Modes" Resource" MUST match the value contains in the 1210-SetOfValues sample.

1210-SetOfValues definition

lwm2m+json version

[{"bn":"/","**n**":"**10/0/14**","**v**":2}]

lwm2m+tlv version

C10 0D 02 ; /10/0/14 2

6.7.3.1.4. LightweightM2M-1.1-int-1203 -- Observation and notification on Object ID:10 related to Power Saving Mode Resources (version 1.1 only)

Test Case Id

LightweightM2M-1.1-int-1203

Test Object

Client and Server

Test Case Description

Sending the observation policy to the device for resources of Cellular network connectivity Object ID:10 in version 1.1

Preconditions

- The Client supports the Configuration C.11 as defined in Annex C
- The Client is registered with the Server in indicating the Object ID:10 is in version 1.1

Test Procedure

A DISCOVER operation is requested by the Server to verify the Object ID:10 is in version 1.1.

The Server establishes an Observation relationship with the Client to acquire conditional notifications about Resources from the Connectivity Management Object Instance:

- Power Saving Modes (ID:13)
- Active Power Saving Modes (ID:14)

Normal flow:

1. DISCOVER (CoAP GET Accept:40) operation is performed by the Server on Object ID:10
2. Server communicates with the Object ID:10 in version 1.1 via WRITE-ATTRIBUTE (CoAP PUT) operations, to set the pmin & pmax Attributes of each targeted Resources (e.g. /10/0/13?pmin=5&pmax=15 and /10/0/14?pmin10&pmax=20)
3. Server sends OBSERVE (CoAP GET operation with Observe Option set to 0) messages for the targeted Resources (ID:13 & ID:14) to activate reporting
4. Client reports requested information with a NOTIFY message (CoAP responses)

Pass-Criteria

1. In test step 1, the Server -- along with the success message 2.05 -- received the information related to Object ID:10

including

</10>;ver="1.1", </10/0>,</10/0/6>,</10/0/11>, </10/0/13>, </10/0/14>

2. In test step 2, the Server receives success messages ("2.04" Changed) related to the WRITE-ATTRIBUTES operations
3. In test step 3, the Server receives success messages ("2.05" Content) along with the initial values of Resource ID:13 and ID:14
4. In test step 4, the Server regularly receives consistent information on the targeted Resources ID:13 & ID:14 of the Object ID:10 Instance 0.

6.7.3.1.5. LightweightM2M-1.1-int-1204 -- Observation and notification on Object ID: 10 related to timers

Test Case Id

LightweightM2M-1.1-int-1204

Test Object

Client and Server

Test Case Description

Sending the observation policy to the device for resources of Cellular network connectivity Object ID:10

Preconditions

- The Client supports the Configuration C.10 as defined in Annex C
- The Client is registered with the Server in indicating the Object ID:10

Test Procedure

A DISCOVER operation is requested by the Server to verify the Object ID: 10.

The Server establishes an Observation relationship with the Client to acquire conditional notifications about Resources from the Connectivity Management Object Instance:

- PSM Timer (ID:4)
- Active Timer (ID:5)
- eDRX parameters for WB-S1 mode(ID:8)
- eDRX parameters for WB-S1 mode(ID:9)

Normal flow:

1. DISCOVER (CoAP GET Accept:40) operation is performed by the Server on Object ID:10
2. Server communicates with the Object ID:10 via WRITE-ATTRIBUTE (CoAP PUT) operations, to set the pmin & pmax Attributes of each targeted Resources (e.g. /10/0/4?pmin=5&pmax=15 and /10/0/5?pmin10&pmax=20 and /10/0/8?pmin=5&pmax=15 and /10/0/9?pmin=10&pmax=20)

3. Server sends OBSERVE (CoAP GET operation with Observe Option set to 0) messages for the targeted Resources (ID:4 & ID:5 & ID:9 & ID:10) to activate reporting
4. Client reports requested information with a NOTIFY message (CoAP responses)

Pass-Criteria

1. In test step 1, the Server -- along with the success message 2.05 -- received the information related to Object ID:10 including

</10>;ver="1.1", </10/0>,</10/0/4>, </10/0/5>, </10/0/6>,</10/0/8>, </10/0/9>,

2. In test step 2, the Server receives success messages ("2.04" Changed) related to the WRITE-ATTRIBUTES operations
3. In test step 3, the Server receives success messages ("2.05" Content) along with the initial values of Resource ID:4 ,ID:5, ID:8 and ID:9
4. In test step 4, the Server regularly receives consistent information on the targeted Resources ID:4 , ID:5, ID:8 and ID:9 of the Object ID:10 Instance 0.

6.7.3.2. APN connection profile ID:11 [1250-1299]

6.7.3.2.1. LightweightM2M-CONMGMT-1.1-int-1250 -- APN configuration

Test Case Id

LightweightM2M-CONMGMT-1.1-int-1250

Test Object

Client and Server

Test Case Description

Creating and enabling a new APN profile

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.10 as defined in Annex C
- Test 104 on registration Update Trigger successfully passed
- Device is switched on and operational
- Client is registered at the LwM2M Server

- Cellular connectivity is established with the parameters given in the APN Connection Profile Object Instance (/11/0).

Test Procedure

A new APN Object is created, then activated. The list of the Active APN Connection Profiles is requested by the Server to the Client

Normal flow:

1. CREATE (COAP POST) operation is performed by the Server targeting APN Connection Profile Object (ID:11) to create a 2nd instance of the APN connection profile Object with a new APN which is not active yet.
2. The Server triggers a Registration Update Trigger for forcing an UPDATE registration from the Client
3. UPDATE (registration) message (COAP POST) is sent from Client to Server including information about the supported Objects and Object Instances and namely the new Instance of the APN Connection Profile Object
4. Server activates the new APN Connection Profile in changing Enable status to True by WRITE operation (COAP PUT /11/1/3)
5. Server reads the list of active APN Connection Profiles by performing a TLV READ targeting Resource ID:11 of Object 10 (/10/0/11)

Pass-Criteria

1. In test step 1., The Server receives a Success message ("2.01" Created) associated to the Instantiation of the APN connection profile Object.
2. In test step 2., the Client receives a Registration Update Trigger request on Resource ID:7 of the Device Object Instance (ID:3)
3. In test step 2., the Server receives the Success message ("2.04" Changed) associated to the Update Registration Request
4. In test step 3, the Server receives an UPDATE (registration) operation from the Client along with the updated list of Object/Object Instances in that Client. This list contains the new Instance of the APN connection profile Object (/11/1)
5. In test step 4, the Server receives the Success message associated with the Server WRITE operation for the new APN activation.
6. In test step 5., the Server receives a Success message ("2.05" Content) along with the list of the active APN containing the new Created APN (11:1)

NOTE: In case the device only supports one active APN profile this test is passed when the new APN profile is activated.

6.7.3.3. WLAN Connectivity ID:12 [1300-1349]

6.7.3.4. Bearer Selection ID:13 [1350-1399]

6.7.3.4.1. LightweightM2M-CONMGMT-1.1-int-1350 -- Bearer Selection

Test Case Id

LightweightM2M-CONMGMT-1.1-int-1350

Test Object

Client and Server

Test Case Description

Controlling bearers using Bearer Selection Object

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the minimum Configuration C.10 as defined in Annex C
- Device is switched on and operational
- Client is registered at the LwM2M server
- The Client supports a Cellular Network Connectivity Object Instance and one instance of an APN Connection Profile Object. Cellular connectivity is established with the parameters given in the APN Connection Profile Object Instance. Also, the Client supports the WLAN Connectivity Object but WLAN radio is not enabled.

Test Procedure

Normal flow:

1. CREATE (COAP POST) operation is performed by the server targeting 13 to create the instance of the Bearer selection object with Preferred Communication Bearer (13/0/0) as WLAN preferred.
2. Server receives success message (2.01 Created)
3. Client SHALL turn on the WLAN radio and use it for connectivity with the Server. Client SHALL send Update to the Server indicating the update in registration as the Client's IP address (and port) has changed.
4. Server checks the status of the WLAN connectivity object by performing a READ on /12/0. The interface SHALL be enabled and running. The Server SHALL verify the values of Enable and Status resources for the same.
5. Server performs WRITE operation on Preferred Communication Bearer (13/0/0) resource and updates its value to 3GPP PS Preferred.
6. Client SHALL turn on the Cellular network connectivity (if not already enabled) and use it for connectivity with the Server. Client SHALL send Update to the Server indicating the update in registration as the Client's IP address (and port) has changed.
7. Server checks the status of the Cellular Network Connectivity object by performing a READ on /10. The interface SHALL be enabled and running.

Pass-Criteria

1. earer Selection Object is allowing the Server to control Client interface for communication

6.7.4. Device Capability Management Object (ID 15) [1500-1599]

Test Cases to fill-up.

6.7.5. Portfolio Object (ID 16) [1600-1699]

6.7.5.1. LightweightM2M-1.1-int-1600 -- Querying the readable resources of object

Test Cases to fill-up.

6.7.5.2. LightweightM2M-1.1-int-1605 -- Setting the writable resources

Test Cases to fill-up.

6.7.5.3. LightweightM2M-1.1-int-1610 -- Basic Observation and notification of a Portfolio Object Instance

Test Cases to fill-up.

6.7.5.4. LightweightM2M-1.1-int-1611 -- Extended Observation and notification of a Portfolio Object Instance

Test Cases to fill-up.

6.7.5.5. LightweightM2M-1.1-int-1620 -- Create Multiple Resource Instances

Test Cases to fill-up.

6.7.5.6. LightweightM2M-1.1-int-1630 -- Create Portfolio Object Instance

Test Case Id

LightweightM2M-1.1-int-1630

Test Object

Client and Server

Test Case Description

Creating a new Instance of the Portfolio Object

Preconditions

- The Client supports the Configuration C.12 as defined in Annex C
- The Client is registered with the Server

Test Procedure

A DISCOVER operations is performed by the Server on the Portfolio Object to retrieve its initial characteristics. A Portfolio CREATE operation is performed to add a new Instance in the LwM2M Client. Additional DISCOVER and READ check the results of the various operations

Normal flow:

1. An initial DISCOVER (CoAP GET Accept:40) operation is performed on Instance 0 of Portfolio Object ID:16
2. A CREATE operation is performed with the Portfolio Object as target and the 1630-SetOfValues as payload in TLV format
3. A new DISCOVER operation is performed on Instance 1 of Portfolio Object ID:16
4. A READ request is performed by the Server on the Portfolio Object (CoAP GET /16).

Pass-Criteria

1. In test step 1, the Success Message ("2.05" Content) is received by the Server along with the payload related to the DISCOVER request and containing :

</16/0/0/>;dim=4

2. In test step 2, a Success message is received by the Server ("2.01" Created) related to the CREATE operation along with the new created Instance ID (SHOULD be /16/1)
3. In test step 3, the Success message ("2.05" Content) is received by the Server along with the payload related to the DISCOVER request and containing :

</16/1/0/>;dim=2

4. In test step 4., the Success message ("2.05" Content) is received by the Server along with a TLV payload related to the READ Request and containing 2 Instances of the Portfolio Object :

- Instance 0 with Identity Resource (ID:0) composed of 4 Instances as defined in Configuration C.12
- Instance 1 with Identity Resource (ID:0) composed of 2 Instances as defined in 1630-SetOfValues below.

1630-SetOfValues definition

lwm2m+tlv version

08 01 2E

80 00 2B

48 00 11 486F737420446576696365204944202332

48 01 14 486F737420446576696365204D6F64656C202332

lwm2m+json version

[{"bn":"/16/1/0/","n":"0","sv":"Host Device ID #2"},

{"n":"1","sv":"Host Device Model #2"}]

6.7.5.7. LightweightM2M-1.1-int-1635 -- Delete Portfolio Object Instance

Test Case Id

LightweightM2M-1.1-int-1635

Test Object

Client and Server

Test Case Description

Deleting all Instances of the Portfolio Object

Preconditions

- The Client supports the Configuration C.12 as defined in Annex C
- The Client is registered with the Server

Test Procedure

A DISCOVER operations is performed by the Server on the Portfolio Object to retrieve its initial characteristics. Portfolio DELETE operations are performed for removing all the Portfolio Object Instances implemented in the LwM2M Client.

Normal flow:

1. An initial DISCOVER (CoAP GET Accept:40) operation is performed on Instance 0 of Portfolio Object ID:16
2. DELETE operations are performed for each Instance of the Portfolio Object
3. A new DISCOVER operation is performed on Portfolio Object ID:16

Pass-Criteria

1. In test step 1, the Success Message ("2.05" Content) is received by the Server along with the payload related to the DISCOVER request and containing at least:

< /16/0>,</16/0/0/>;dim=4

2. In test step 2, successive Success messages are received by the Server ("2.04" Deleted) related to each DELETE operation targeting a Portfolio Object Instance

3. In test step 3, the Success message ("2.05" Content) is received by the Server along with the payload related to the DISCOVER request and containing: </16>

(Object ID16 present but without any Instance implemented in the Client)

6.7.6. BinaryAppDataContainer Object (ID 19) [1900-1999]

paragraph

6.7.6.1. LightweightM2M-1.1-int-1900 -- Observation and notification on Object ID: 19

Test Case Id

LightweightM2M-1.1-int-1900

Test Object

Client and Server

Test Case Description

Sending the observation policy to the device for resources of BinaryAppDataCont Object ID:19

Preconditions

- The Client supports the Configuration C.15 as defined in Annex C
- The Client is registered with the Server in indicating the Object ID:19
- Instance 0 is used to send application data from LwM2M client to LwM2M server.
- Instance 1 is used to send application data from LwM2M server to LwM2M client.

Test Procedure

A DISCOVER operation is requested by the Server to verify the Object ID:19

The Server establishes an Observation relationship with the Client to acquire conditional notifications about Resources from the Binary AppData Container Instance 0:

- Data (ID: 0)

Normal flow:

1. DISCOVER (CoAP GET Accept:40) operation is performed by the Server on Object ID:19
2. Server communicates with the Object ID:19 via WRITE-ATTRIBUTE (CoAP PUT) operations, to set the pmin & pmax Attributes of the targeted Resource (e.g. /19/0/0?pmin=300&pmax=3600)
3. Server sends OBSERVE (CoAP GET operation with Observe Option set to 0) messages for the targeted Resources(ID:0) to activate reporting
4. Client reports requested information with a NOTIFY message (CoAP responses)

Pass-Criteria

1. In test step 1, the Server -- along with the success message 2.05 -- received the information related to Object ID:19 including

</19>;ver="1.1", </19/0>,</19/0/0>

2. In test step 2, the Server receives success messages ("2.04" Changed) related to the WRITE-ATTRIBUTES operations
3. In test step 3, the Server receives success messages ("2.05" Content) along with the initial values of Resource ID: 0
4. In test step 4, the Server regularly receives consistent information on the targeted Resources ID: 0 Instance 0.

6.7.6.2. LightweightM2M-1.1-int-1901 -- Setting the writable Resource Data

Test Case Id

LightweightM2M-1.1-int-1901

Test Object

Client and Server

Test Case Description

Setting the writable Resource Data (ID:0) of Binary AppData Container Instance 1 and verifying the triggered actions produce the expected results

Tool

n/a

Test code

n/a

Preconditions

- The Client supports the Configuration C.15 as defined in Appendix C
- The Client is registered with the Server in indicating the Object ID:19
- Instance 0 is used to send application data from LwM2M client to LwM2M server.
- Instance 1 is used to send application data from LwM2M server to LwM2M client.

Test Procedure

A WRITE operation is performed by the Server to update a valid value in the Client in setting the Data Resource.

The various Resources of this Binary AppData Container Objects MUST be set as expected by the Client.

Normal flow:

1. A WRITE (CoAP PUT) operation with a valid value is performed by the Server on the Data Resource (/19/1/0) of the Binary AppData Container Object Instance 1

2. The Server READs (CoAP GET) the Binary AppData Container Object Instance 1 to get the values of the Data (/19/1/0)

Pass-Criteria

1. In test step 1, the Server receives the success message "2.04" associated with the WRITE request.
2. In test step 2, the Server receives the success message "2.05" along with the Data Resources values.
3. In test step2, the queried value of Data resource is the same with that in Step1.

6.7.7. Event Log Object (ID 20) [2000-2099]

6.7.7.1. LightweightM2M-1.1-int-2000 -- Control the Log Function

Test Case Id

LightweightM2M-1.1-int-2000

Test Object

Client and Server

Test Case Description

Setting the writable Resource LogStart (ID:4011) of Event Log and verifying the triggered actions produce the expected results

Preconditions

- The Client supports the Configuration C.16 as defined in Appendix C
- The Client is registered with the Server in indicating the Object ID:20

Test Procedure

A WRITE operation is performed by the Server to update a valid value in the Client in setting the LogStart Resource.

The various Resources of this Event Log Objects MUST be set as expected by the Client.

Normal flow:

1. A WRITE (CoAP PUT) operation with a valid value is performed by the Server on the LogStart Resource (/20/0/4011) of the Event Log Object
2. The Server READs (CoAP GET) the Event Log Object to get the values of the LogStart (/20/0/4011)

Pass-Criteria

1. In test step 1, the Server receives the success message "2.04" associated with the WRITE request.
2. In test step 2, the Server receives the success message "2.05" along with the LogStart Resources values.
3. In test step2, the queried value of LogStart resource is the same with that in Step1.

6.7.7.2. LightweightM2M-1.1-int-2001 -- Querying the readable resources

Test Case Id

LightweightM2M-1.1-int-2001

Test Object

Client and Server

Test Case Description

Querying Information on the Firmware Update Object Instance for determining the current states and supported characteristics of the Client for such capability.

Preconditions

- The Client supports the Configuration C.16 as defined in Annex C
- The Client is registered with the Server in indicating the Object ID:20

Test Procedure

A READ operation from the Server on the EventLog Object Instance requesting TLV format has been received by the client.

Normal flow:

1. READ (CoAP GET) operation is performed on the EventLog Object Instance

Pass-Criteria

1. In test step 1 :the Server receives the success message (2.05 Content) associated to its READ request
2. In step 1: the values returned by the Client in TLV format are consistent with the Configuration C 16 used for that test (LogStatus (ID:4013), LogClass (ID:4010), and LogData(4014))

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
OMA-ETS-LightweightM2M-V1_0_1-20170926-A	26 Sep 2017	Status changed to Approved by TP, TP Ref # OMA-TP-2017-0039-INP_LightweightM2M-V1_0_ETS_for_Final_Approval
OMA-ETS-LightweightM2M-V1_0_2-20180815-A	15 Aug 2018	Status changed to Approved by DM, Doc Ref # OMA-DM&SE-2018-0088-INP_LightweightM2M_V1_0_ETS_for_Final_Approval
OMA-ETS-LightweightM2M-V1_2-20231003-A	03 Oct 2023	Document Approved by DMSO on October 3rd and ratified by the OMA BoD on Oct 16th.
OMA-ETS-LightweightM2M-V1_2_1-20241203-C	12 Mar 2024	Document Approved by DMSO on March 12th.

Appendix B. Additional Information

B.1 Example of Test Configuration and Setup

The following hardware components were part of a test setup.

- M2M device equipped with LwM2M client
- Computer that runs the browser interface to the LwM2M Server component
- Server that runs the LwM2M Server software
- USIM provisioned for use on the network.
- External appliance which is connected directly to M2M device (e.g. light, temperature sensor, motor).

In addition the demonstration setup SHALL include the following software components:

- Measurement software which enables to see e.g. LwM2M messages and CoAP messages.

The software SHALL enable

- GUI to trigger the chosen test cases
- To see the flow of information between client and Server, e.g. on LwM2M protocol level and on CoAP transport level.

Appendix C. LwM2M Configurations

When a Test Case references a certain Configuration of that Annex, it means the features of that configuration are the minimum ones which MUST be supported by LwM2M Clients & Servers to pass that test; a richer configuration (superset) MAY be used instead, as long as it still contains the minimum features.

C.1 Basic Configuration 1

PSK Mode support, no SMS support

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11/0","v":0},

{"n":"3/0/16","vs":"U"}]

Server Object ID:1 implements the mandatory Executable Resource ID:8 - Registration Update Trigger

Device Object ID:3 implements the mandatory Executable Resource ID:4 - Reboot

C.2 Basic Configuration 2

PSK Mode & SMS supports

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/6","v":3},

{"n":"0/0/7","vd":<KIc,KID,SPI,TAR>}, // opaque in Base64

{"n":"0/0/8","vd":<SMS Secret Keys>}, // opaque in Base64

{"n":"0/0/9","vs":<MSISDN>},

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11/0","v":0},

{"n":"3/0/16","vs":"US"}]

Server Object ID:1 implements the mandatory Executable Resource ID:8 - Registration Update Trigger

Device Object ID:3 implements the mandatory Executable Resource ID: - Reboot

C.3 Configuration 3

This configuration is a superset of Configuration C.1.

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/2","v":1},

{"n":"1/0/3","v":10},

{"n":"1/0/5","v":86400},

{"n":"1/0/6","vd":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11","v":0},

{"n":"3/0/16","vs":"U"}]

Server Object ID:1 implements the mandatory Executable Resource ID:8 – Registration Update Trigger

Server Object ID:1 implements the optional Executable Resource ID:4 – DisableDevice Object ID:3 implements the mandatory Executable Resource ID:4 – Reboot

C.4 Configuration 4

Configuration 4 is a superset of Configuration 3 and can be used instead ; it implements the optional Resources related to external or internal battery which the voltage could vary.

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/2","v":1},

{"n":"1/0/3","v":10},

{"n":"1/0/5","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/6/0","v":1}, // Internal Battery

{"n":"3/0/6/1","v":2}, // External Battery

{"n":"3/0/7/0","v":0}, // Internal Battery mV

{"n":"3/0/7/1","v":0}, // External Battery mV

{"n":"3/0/8/0","v":0}, // Internal Battery mA

{"n":"3/0/8/1","v":0}, // External Battery mA

{"n":"3/0/9","v":0}, // Internal Battery level

{"n":"3/0/11","v":0},

{"n":"3/0/16","vs":"U"}]

Server Object ID:1 implements the mandatory Executable Resource ID:8 - Registration Update Trigger

Server Object ID:1 implements the optional Executable Resource ID:4 - Disable

Device Object ID:3 implements the mandatory Executable Resource ID:4 - Reboot

C.5 Configuration 5

This configuration is a superset of Configuration C.1 and can be used instead; it implements the full Connectivity Monitoring Object ID:4.

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/2","v":1},

{"n":"1/0/3","v":10},

{"n":"1/0/5","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11","v":0},

{"n":"3/0/16","vs":"U"},

{"n":"4/0/0","v":<Network Bearer>},

{"n":"4/0/1/0","v":<Available Network Bearer 0>},

{"n":"4/0/1/1","v":<Available Network Bearer 1>},

{"n":"4/0/2","v":<Radio Signal Strength>},

{"n":"4/0/3","v":<Link Quality>},

{"n":"4/0/4/0","vs":<IP Address 0>},

{"n":"4/0/4/1","vs":<IP Address 1>},

{"n":"4/0/5/0","vs":<Router IP Address 0>},

{"n":"4/0/5/1","vs":<Router IP Address 1>},

{"n":"4/0/6","v":<Link Utilization>},

{"n":"4/0/7/0","vs":<APN 0>},

{"n":"4/0/7/1","vs":<APN 1>},

{"n":"4/0/8","v":<Cell ID>},

{"n":"4/0/9","v":<SMNC>},

{"n":"4/0/10","v":<SMCC>}]

Server Object ID:1 implements the mandatory Executable Resource ID:8 – Registration Update Trigger

Device Object ID:3 implements the mandatory Executable Resource ID:4 – Reboot

C.6 Configuration 6

Configuration 6 is a superset of Configuration 4 and can be used instead; the configuration C.4, has been completed with all the optional resources of the Device Object.

[{ "bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/2","v":1},

{"n":"1/0/3","v":10},

{"n":"1/0/5","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/6/0","v":1}, // Internal Battery

{"n":"3/0/6/1","v":2}, // External Battery

{"n":"3/0/7/0","v":<Internal Battery Voltage mV>},

{"n":"3/0/7/1","v":<External Battery Voltage mV >},

{"n":"3/0/8/0","v":<Internal Battery Current mA>},

{"n":"3/0/8/1","v":<External Battery Current mA >},

{"n":"3/0/9","v":< Internal Battery Level>},

{"n":"3/0/10","v":<Memory Free>},

{"n":"3/0/11","v":<Error Code>},

{"n":"3/0/13","t":<Current Time >},

{"n":"3/0/14","vs":<UTC Offset>},

{"n":"3/0/15","vs":<TimeZone >},

{"n":"3/0/16","vs":U},

{"n":"3/0/17","vs":<Device Type>},

{"n":"3/0/18","vs":<Hardware Version>},

{"n":"3/0/19","vs":<Software Version>},

{"n":"3/0/20","vs":<Battery Status>},

{"n":"3/0/21","vs":<Memory Total>},

{"n":"3/0/22","vlo":<ExtDevInfo>}]

Server Object ID:1 implements the mandatory Executable Resource ID:8 - Registration Update Trigger

Server Object ID:1 implements the optional Executable Resource ID:4 - Disable

Device Object ID:3 implements the mandatory Executable Resource ID:4 - Reboot

Device Object ID:3 implements the optional Executable Resource ID:5 - Factory Reset

Device Object ID:3 implements the optional Executable Resource ID:12 -- Reset Error Code

C.7 Configuration 7

This configuration is a superset of Configuration C.1 and can be used instead ; it implements the full Location Object ID:6.

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/2","v":1},

{"n":"1/0/3","v":10},

{"n":"1/0/5","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11","v":0},

{"n":"3/0/16","vs":"U"},

{"n":"6/0/0","v":43.5723},

{"n":"6/0/1","v":153.21760},

{"n":"6/0/2","v":140},

{"n":"6/0/3","v":100},

{"n":"6/0/4","vd": "0"}, // opaque 3GPP-TS- 23.032 Base64 representation

{"n":"6/0/5","t":1367491215}, // time

{"n":"6/0/6","v": 3}]

Server Object ID:1 implements the mandatory Executable Resource ID:8 - Registration Update Trigger

Device Object ID:3 implements the mandatory Executable Resource ID:4 - Reboot

C.8 Configuration 8

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/2","v":1},

{"n":"1/0/3","v":10},

{"n":"1/0/5","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11","v":0},

{"n":"3/0/16","vs":"U"},

{"n":"5/0/0","vs":<Firmaware Package>}, // opaque in Base64

{"n":"5/0/1","vs":<Package URI>},

{"n":"5/0/3","v":<State>},

{"n":"5/0/5","v":<Update Result>},

{"n":"5/0/6","vs":<Package Name>},

{"n":"5/0/7","vs":<Package Version>},

{"n":"5/0/8","v":<Firmware Update Protocol Support> }, // CoaP only if absent

{"n":"5/0/9","v":<Firmware Update Delivery Method> }]

Server Object ID:1 implements the mandatory Executable Resource ID:8 - Registration Update Trigger

Server Object ID:1 implements the optional Executable Resource ID:4 - Disable

Device Object ID:3 implements the mandatory Executable Resource ID:4 - Reboot

Device Object ID:5 implements the mandatory Executable Resource ID:2 - Update

C.9 Configuration 9

This configuration is a superset of Configuration C.1 and can be used instead ; it implements the full Connectivity Statistics Object ID:7

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/2","v":1},

{"n":"1/0/3","v":10},

{"n":"1/0/5","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11","v":0},

{"n":"3/0/16","vs":"U"},

{"n":"7/0/0","v":0},

{"n":"7/0/1","v":0},

{"n":"7/0/2","v":0},

{"n":"7/0/3","v":0},

{"n":"7/0/4","v": 0},

{"n":"7/0/5","v":0},

{"n":"7/0/8","v":0}]

Server Object ID:1 implements the mandatory Executable Resource ID:8 – Registration Update Trigger

Device Object ID:3 implements the mandatory Executable Resource ID:4 – Reboot

Device Object ID:7 implements the mandatory Executable Resource ID:6 – Start

Device Object ID:7 implements the mandatory Executable Resource ID:7 – Stop

C.10 Basic Connectivity Management Configuration 10

PSK Mode support, no SMS support.

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11/0","v":0},

{"n":"3/0/16","vs":"U"},

{"n":"10/0/4","v":3600},

{"n":"10/0/5","v":600},

{"n":"10/0/6","v":40},

{"n":"10/0/8","v": 20.48},

{"n":"10/0/9","v": 51.2},

{"n":"11/0/0","vs":<Profile Name>},

{"n":"11/0/3","b":true},

{"n":"12/0/0","vs":<wlan identifier>"},

{"n":"12/0/1","v":0},

{"n":"12/0/3","v":0},

{"n":"12/0/4","vs":<MAC address>},

{"n":"12/0/8","v":0},

{"n":"12/0/4","v":0},

{"n":"12/0/14","v":0},

{"n":"12/0/15","v":0}]

C.11 Basic Connectivity Management Configuration 11 supporting Object ID:10 version 1.1

PSK Mode support, no SMS support.

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11/0","v":0},

{"n":"3/0/16","vs":"U"},

{"n":"10/0/4","v":3600},

{"n":"10/0/5","v":600},

{"n":"10/0/6","v":40},

{"n":"10/0/8","v": 20.48},

{"n":"10/0/9","v": 51.2},

{"n":"10/0/11","vlo":"11:0"},

{"n":"10/0/13","v":3},

{"n":"10/0/14","v":1},

{"n":"11/0/0","vs":<Profile Name>},

{"n":"11/0/3","vb":true},

{"n":"12/0/0","vs":<wlan identifier>"},

{"n":"12/0/1","v":0},

{"n":"12/0/3","v":0},

{"n":"12/0/4","vs":<MAC address>},

{"n":"12/0/8","v":0},

{"n":"12/0/4","v":0},

{"n":"12/0/14","v":0},

{"n":"12/0/15","v":0}]

C.12 Configuration 12

This configuration is a superset of Configuration C.1 and can be used instead; it implements the Basic Portfolio Object ID:16 (Identity Resource with 4 Instances).

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/2","v":1},

{"n":"1/0/3","v":10},

{"n":"1/0/5","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11","v":0},

{"n":"3/0/16","vs":"U"},

{"n":"16/0/0/0","vs":"Host Device ID #1"},

{"n":"16/0/0/1","vs":" Host Develce Manufacturer #1"},

{"n":"16/0/0/2","vs":" Host Device Model #1"},

{"n":"16/0/0/3","vs":" Host Device Software Version #1"}]

Server Object ID:1 implements the mandatory Executable Resource ID:8 - Registration Update Trigger

Device Object ID:3 implements the mandatory Executable Resource ID:4 - Reboot

C.13 Bootstrap Server Contact Configuration 13

PSK Mode support, no SMS support.

[{"bn":"/","n":"0/1/0","vs": <LwM2M Bootstrap Server URI>},

{"n":"0/1/1","vb":true},

{"n":"0/1/2","v":0},

{"n":"0/1/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/1/4","vs": <n/a> },

{"n":"0/1/5","vd": <Secret Key>}, // opaque in Base64

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11/0","v":0},

{"n":"3/0/16","vs":"U"}]

The Mandatory Object Server (ID:1) is supported by the Client but has no Instance in that Configuration

C.14 Multi-Servers Context Initial Configuration 14

PSK Mode support, no SMS support.

[{"bn":"/","n":"0/1/0","vs": <LwM2M Bootstrap Server URI>},

{"n":"0/1/1","vb":true},

{"n":"0/1/2","v":0},

{"n":"0/1/3","vd":<PSK Identity #0>}, // opaque in Base64

{"n":"0/1/4","vs": <n/a> },

{"n":"0/1/5","vd": <Secret Key #0>}, // opaque in Base64

// Server Account #1

{"n":"0/0/0","vs": <LwM2M Server URI #1>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity #1>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key #1>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/2","v":1},

{"n":"1/0/3","v":10},

{"n":"1/0/5","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

// Server Account #2

{"n":"0/2/0","vs": <LwM2M Server URI #2>},

{"n":"0/2/1","vb":false},

{"n":"0/2/2","v":0},

{"n":"0/2/3","vd":<PSK Identity #2>}, // opaque in Base64

{"n":"0/2/4","vs": <n/a> },

{"n":"0/2/5","vd": <Secret Key #2>}, // opaque in Base64

{"n":"0/2/10","v":2},

{"n":"1/1/0","v":2},

{"n":"1/1/1","v":86400},

{"n":"1/1/2","v":1},

{"n":"1/1/3","v":10},

{"n":"1/1/5","v":86400},

{"n":"1/1/6","vb":false},

{"n":"1/1/7","vs":"U"},

// Access Control Object : Server #1 has full access, Server #2 no access

{"n":"2/0/0","v":1},

{"n":"2/0/1","v":0},

{"n":"2/0/2/1","v":15}, // Server #1 full access

{"n":"2/0/3","v":1},

// Access Control Object : Server #2 has full access, Server #1 no access

{"n":"2/0/0","v":1},

{"n":"2/0/1","v":1},

{"n":"2/0/2/2","v":15}, // Server #2 full access

{"n":"2/0/3","v":2},

// Access Control Object : Object Device Server #2 & #1 have full access (i.e. RWE)

{"n":"2/0/0","v":3},

{"n":"2/0/1","v":0},

{"n":"2/0/2/2","v":15},

{"n":"2/0/3","v":<MAX_ID>},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11/0","v":0},

{"n":"3/0/16","vs":"U"}]

// C14

C.15 Binary AppData Container Configuration 15

This configuration is a superset of Configuration C.1 and can be used instead; it implements the Binary AppData Container Object ID: 19 (Identity Resource with 4 Instances).

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a>},

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/2","v":1},

{"n":"1/0/3","v":10},

{"n":"1/0/5","v":86400},

{"n":"1/0/6","vb": false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs:<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11","v":0},

{"n":"3/0/16","vs":"U"},

{"n":"19/0/0","vd":"
<InNlcnZpY2VJZCI6Ik1ldGVyIiwNCiJzZXJ2aWNlRGF0YSI6ew0KImN1cnJlbnRSZWFkaW5nIjoiNDYuMyIsDQoic2lnbmFsU3RyZW5ndGgiOjE2LA0KImRhaWx5QWN0aXZpdHlUaW1lIjo1NzA2DQo=>, // opaque in Base64

{"n":"19/1/0","vs":" < n/a>"}]

Server Object ID: 1 implements the mandatory Executable Resource ID: 8 – Registration Update Trigger

Device Object ID: 3 implements the mandatory Executable Resource ID: 4 – Reboot

C.16 Event Log Configuration 16

This configuration is a superset of Configuration C.1 and can be used instead; it implements the Event Log Object ID: 20 (Identity Resource with 4 Instances).

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a>},

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/2","v":1},

{"n":"1/0/3","v":10},

{"n":"1/0/5","v":86400},

{"n":"1/0/6","vb": false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11","v":0}, {"n":"3/0/16","vs":"U"},

{"n":"20/0/4010","v":0 },

{"n":"20/0/4011","vs":"0,100" },

{"n":"20/0/4013","v":3 }

{"n":"20/0/4014","vd":"61-7C-E3-01-C1-11-00-00-05-00-60-18-18-18-0C-00-01-41-06-00-02-00-40-0C-00-00-00-00"}]

// opaque in Base64

Server Object ID: 1 implements the mandatory Executable Resource ID: 8 - Registration Update Trigger

Device Object ID: 3 implements the mandatory Executable Resource ID: 4 - Reboot

C.17 Basic Configuration 17

Note: The referenced certificates & keys are provided in a separate ZIP file.

This configuration is a modification of the "Basic Configuration 1" file, which supports certificate-based authentication.

[{"bn":"/","n":"0/0/0","vs": coaps://server.example.com}, // The certificate was created for server.example.com.

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":2}, // certificate mode

{"n":"0/0/3","vs":<Client Certificate >}, // Contains the client.crt from ZIP archive

{"n":"0/0/4","vs": <Server Certificate> }, // Contains the server.crt from ZIP archive

{"n":"0/0/5","vs": <Client Private Key>}, // Contains the client.key from ZIP archive

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11/0","v":0},

{"n":"3/0/16","vs":"U"}]

Server Object ID:1 implements the mandatory Executable Resource ID:8 - Registration Update Trigger

Device Object ID:3 implements the mandatory Executable Resource ID:4 - Reboot

C.18 Basic Configuration 18

Note: The referenced certificates & keys are provided in a separate ZIP file.

This configuration is a modification of the "Basic Configuration 17" file, which supports certificate-based authentication. However, the LwM2M Server URI is set to a different value.

[{"bn":"/","n":"0/0/0","vs": coaps://server-fail.example.com}, // The certificate was created for server.example.com.

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":2}, // certificate mode

{"n":"0/0/3","vs":<Client Certificate >}, // Contains the client.crt from ZIP archive

{"n":"0/0/4","vs": <Server Certificate> }, // Contains the server.crt from ZIP archive

{"n":"0/0/5","vs": <Client Private Key>}, // Contains the client.key from ZIP archive

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11/0","v":0},

{"n":"3/0/16","vs":"U"}]

Server Object ID:1 implements the mandatory Executable Resource ID:8 - Registration Update Trigger

Device Object ID:3 implements the mandatory Executable Resource ID:4 - Reboot

C.19 Basic Configuration 19

This configuration is used for CoAP over TLS using certificate-based authentication.

Note: The referenced certificates & keys are provided in a separate ZIP file.

[{"bn":"/","n":"0/0/0","vs": coaps+tcp://server.example.com}, // The certificate was created for server.example.com.

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":2}, // certificate mode

{"n":"0/0/3","vs":<Client Certificate >}, // Contains the client.crt from ZIP archive

{"n":"0/0/4","vs": <Server Certificate> }, // Contains the server.crt from ZIP archive

{"n":"0/0/5","vs": <Client Private Key>}, // Contains the client.key from ZIP archive

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"T"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11/0","v":0},

{"n":"3/0/16","vs":"T"}]

Server Object ID:1 implements the mandatory Executable Resource ID:8 - Registration Update Trigger

Device Object ID:3 implements the mandatory Executable Resource ID:4 - Reboot

C.20 Basic Configuration 20

This configuration is used for CoAP over TLS using pre-shared-key authentication.

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":0},

{"n":"0/0/3","vd":<PSK Identity>}, // opaque in Base64

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <Secret Key>}, // opaque in Base64

{"n":"0/0/10","v":1},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"T"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11/0","v":0},

{"n":"3/0/16","vs":"T"}]

Server Object ID:1 implements the mandatory Executable Resource ID:8 - Registration Update Trigger

Device Object ID:3 implements the mandatory Executable Resource ID:4 - Reboot

C.21 Basic Configuration 21

This configuration is used for EST-based bootstrapping.

Note: The referenced certificates & keys are provided in a separate ZIP file. With EST the client certificate and the client private key are not configured into resources 3 and 5 but instead available to EST directly.

The client certificate and the client private key to authenticate to the LwM2M Bootstrap-Server can be found in client.crt and in client.key of the ZIP archive.

[{"bn":"/","n":"0/0/0","vs": <LwM2M Bootstrap-Server URI>},

{"n":"0/0/1","vb":true}, // LwM2M Bootstrap-Server

{"n":"0/0/2","v":4}, // EST mode

{"n":"0/0/3","vs":<Client Certificate >}, // Contains the client.crt from ZIP archive

{"n":"0/0/4","vs": <Server Certificate> }, // Contains the server.crt from ZIP archive

{"n":"0/0/5","vs": <Client Private Key>}, // Contains the client.key from ZIP archive

{"n":"0/0/10","v":0}, // Short Server ID

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11/0","v":0},

{"n":"3/0/16","vs":"U"}]

C.22 Basic Configuration 22

OSCORE Security

[{"bn":"/","n":"0/0/0","vs": <LwM2M Server URI>},

{"n":"0/0/1","vb":false},

{"n":"0/0/2","v":3},

{"n":"0/0/3","vd":<n/a>},

{"n":"0/0/4","vs": <n/a> },

{"n":"0/0/5","vd": <n/a>},

{"n":"0/0/10","v":1},

{"n":"0/0/17","v":"/21/0"},

{"n":"1/0/0","v":1},

{"n":"1/0/1","v":86400},

{"n":"1/0/6","vb":false},

{"n":"1/0/7","vs":"U" or "T"},

{"n":"3/0/0","vs":<Manufacturer Name>},

{"n":"3/0/1","vs":<Model Name>},

{"n":"3/0/2","vs":<Serial Number>},

{"n":"3/0/3","vs":<Firmware Version>},

{"n":"3/0/11/0","v":0},

{"n":"3/0/16","vs":"U"},

{"n":"21/0/0","v":<Master Salt Key>}, // opaque in Base64

{"n":"21/0/1","v":<Sender ID>},

{"n":"21/0/2","v":<Receiver ID>}

]

C.23 Bootstrap Server Contact Configuration 23

OSCORE Security

[{"bn":"/0/0/","n":"0","vs": <LwM2M Bootstrap Server URI>},

{"n":"1","vb":true},

{"n":"2","v":3},

{"n":"3","vd":<n/a>},

{"n":"4","vs": <n/a>},

{"n":"5","vd": <n/a>},

{"n":"10","v":<n/a>},

{"n":"17","v":"/21/0"},

{"bn":"/3/0/", "n":"0","vs":<Manufacturer Name>},

{"n":"1","vs":<Model Name>},

{"n":"2","vs":<Serial Number>},

{"n":"3","vs":<Firmware Version>},

{"n":"11/0","v":0},

{"n":"16","vs":"U"},

{"bn":"/21/0/","n":"0","v":<Master Salt Key>}, // opaque in Base64

{"n":"1","v":<Sender ID>},

{"n":"2","v":<Receiver ID>}

]

C.24 Basic Configuration 24 using OSCORE

[{"bn":"/0/1/","n":"0","vs": <LwM2M Server URI>},

{"n":"1","vb":false},

{"n":"2","v":3},

{"n":"3","vd":<n/a>},

{"n":"4","vs": <n/a>},

{"n":"5","vd": <n/a>},

{"n":"10","v":1},

{"n":"17","v":"/21/1"},

{"bn":"/1/1/","n":"0","v":1},

{"n":"1","v":86400},

{"n":"6","vb":false},

{"n":"7","vs":"U"},

{"bn":"/3/0/", "n":"0","vs":<Manufacturer Name>},

{"n":"1","vs":<Model Name>},

{"n":"2","vs":<Serial Number>},

{"n":"3","vs":<Firmware Version>},
{"n":"11/0","v":0},
{"n":"16","vs":"U"},
{"bn":"/21/1/","n":"0","v":<Master Salt Key>}, // opaque in Base64
{"n":"1","v":<Sender ID>},
{"n":"2","v":<Receiver ID>}
]

C.25 Bootstrap Server Contact Configuration 25 MQTT Transport

MQTT, PSK Mode support, no SMS support; similar to Configuration 13, but using MQTT transport for bootstrapping

The Mandatory Object Server (ID:1) is supported by the Client but has no Instance in this Configuration

[{"bn":"/0/1/","n":"0","vs": <MQTT Server/Broker URI to use for Bootstrap>},
{"n":"1","vb":true},
{"n":"2","v":0},
{"n":"3","vd":<PSK Identity>}, // opaque in Base64
{"n":"0/1/4","v": "<n/a>},
{"n":"5","vd": <Secret Key>}, // opaque in Base64
{"n":"26","vs": "/24/0"> },
{"n":"27","vs": "/23/0"> },
{"bn":"/3/0/", "n":"0","vs":<Manufacturer Name>},
{"n":"1","vs":<Model Name>},
{"n":"2","vs":<Serial Number>},
{"n":"3","vs":<Firmware Version>},
{"n":"11/0","v":0},
{"n":"16","vs":"M"},
{"bn":"/23/0/","n":"0","vs":<Key Identifier>},
{"n":"1","v":0},
{"n":"2","vs":<Secret Key>},
{"bn":"/24/0/","n":"3","vb":true},
{"n":"6","vs":<Client Identifier>}]

C.26 Basic Configuration 26 MQTT Transport

[{"bn":"/0/1/","n":"0","vs": <MQTT Server/Broker URI>},

{"n":"1","vb":false},

{"n":"2","v":0},

{"n":"3","vd":<PSK Identity>}, // opaque in Base64

{"n":"4","v": "<n/a>},

{"n":"5","vd": <Secret Key>}, // opaque in Base64

{"n":"10","v":1},

{"n":"26","vs": "/24/1"> },

{"n":"27","vs": "/23/1"> },

{"bn":"/1/1/","n":"0","v":1},

{"n":"1","v":86400},

{"n":"6","vb":false},

{"n":"7","vs":"M"},

{"bn":"/3/0/", "n":"0","vs":<Manufacturer Name>},

{"n":"1","vs":<Model Name>},

{"n":"2","vs":<Serial Number>},

{"n":"3","vs":<Firmware Version>},

{"n":"11/0","v":0},

{"n":"16","vs":"M"},

{"bn":"/23/1/","n":"0","vs":<Key Identifier>},

{"n":"1","v":0},

{"n":"2","vs":<Secret Key>},

{"bn":"/24/1/","n":"6","vs":<Client Identifier>}

]

Appendix D. Core Test Coverage

New Appendix

Appendix E. Enabler Validation Plan

The validation of this enabler will be made based on the successful execution of test cases defined in the [LWM2M-ETS].

A summary of test results will be the evidence that those test cases were executed. The lack of open Problem Reports will be the evidence the test cases were successful executed and that any issue found whether on test specifications, technical specifications or other was correctly addressed.

E.1 Entry Criteria for TestFest

The following tests need to be performed and passed by implementations by members wishing to participate in the TestFest. This ensures minimal requisite capability of the implementations. The tests are defined in the ETS [LWM2M-ETS] and any special comments are noted.

Test Case Id	Special Conditions
LightweightM2M_1.1-int-101	
LightweightM2M_1.1-int-102	
LightweightM2M_1.1-int-201	
LightweightM2M_1.1-int-203	

Table: Listing of Tests for Entry Criteria for TestFest

E.2 Enabler Validation Test Cases

The following table should list the set of tests that are used for enabler validation. Not applicable in this case as the enabler has been already agreed.

Test Case Id	ETR Requirement Id/Feature Key	ETR Status	Notes

Table: Enabler Validation Test Cases