Lightweight M2M – Portfolio Object (LWM2M Object – PortfolioObj)

Candidate Version 1.0 – 06 Sep 2016

Open Mobile Alliance

OMA-TS-LWM2M_PortfolioObj-V1_0-20160906-C

Contents

Figures

Tables

1. Scope

This document defines the technical specification for an Object to be used in conjunction with the Lightweight M2M enabler in order to extend the data storage capability of other Object Instances in the system.

Moreover, such data extension MAY come with the support of new functionalities as strong data authentication (which confidence can be granted to a data likely to represent a personal ID).

2. References

2.1 Normative References

[LWM2M] "OMA LightweightM2M", Version 1.0, Open Mobile Alliance™, URL:http://www.openmobilealliance.org/

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997, URL:http://www.ietf.org/rfc/rfc2119.txt

2.2 Informative References

[OMADICT] "Dictionary for OMA Specifications", Version 2.9, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_9, URL:http://www.openmobilealliance.org/

[GSMA] "GSM Association CLP.03 –IOT Device Connection Efficiency Guidelines Version 1.1 30 January 2015"

[GP] "Global Platform SE configuration v1.0"

[RFC4493] The AES-CMAC Algorithm (2006)

3. Terminology and Conventions

3.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

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

3.2 Definitions

Tamper-resistant component	A hardware component used to store and process private or sensitive information, such as private keys. To prevent an attacker from retrieving or modifying the information, the component is designed so that the information is not accessible through external means and can be accessed only by the embedded software

3.3 Abbreviations

OMA	Open Mobile Alliance

4. Introduction

Some LWM2M Applications may require extended data storage capacity over what is available with existing LWM2M Object Instances. This storage capacity extension can be used to manage information from a device that is closely related to a LWM2M Client even if this information is not directly used by the LWM2M Client itself (e.g. the identity of a Host Device having an embedded module containing a LWM2M Client).

Moreover, these extended data may be associated with specific constraints. For example, some data could be used to univocally identify a person or a device; in that case appropriate authentication services MUST be proposed to fulfil that need.

The Portfolio Object specification defines a mechanism for extending the data storage capability of other Object Instances in the LWM2M system, as well as the services which may be used to authenticate and to protect privacy of data contained in those extensions. In addition, a data encryption service is also defined.

5. Portfolio Object Functionality

The Portfolio Object offers the possibility to extend the data storage capability of another LWM2M Object instance by attaching one of the Portfolio Object Instances to the set of resources of the original LWM2M Object Instance.

Moreover, a second capability of the Portfolio Object is to provide security services for applications associated with this data extension. These services can be used to authenticate identities and to encrypt data.

5.1 Portfolio Data Storage Extension

A Portfolio Object Instance and its Resources MUST be fully dedicated to a given LWM2M Object Instance.

Identity Resource (Resource 0 of the Portfolio Object definition) is a mandatory resource that maybe instantated multiple times.

Appendix B provides one example targeting the GSMA CLP0.3 requirement: a Portfolio Object Instance (e.g. Instance 0) is attached to the LWM2M Device Object Instance (LWM2M Object '3'). This Portfolio Object Instance defines four *Identity* resource instances fitting with the GSMA requirements related to the Host Device information.

5.2 Portfolio Services

5.2.1 Resources description

The Portfolio Object defines three Resources to support Services that can be used in conjunction with the *Identity* resource: GetAuthData, AuthData, AuthStatus.

1. ***GetAuthData*:** defines a executable Resource which is used to trigger the process of generating data according to the set of arguments provided with the LWM2M Server request.

    - *Argument 0*: 'Service': the type of Service required. Two Services are specified in section 5.2.2: Data Authentication and Data encryption. Services

        0: SRV_AUTHENT: Data Authentication Service

        1: SRV_ENCRYPT: Data Encryption Service
    - *Argument 1*: Identity Instance ID: determins which *Identity* Resource Instance is concerned by the *GetAuthData* request
    - *Argument 2:* Challenge: a random value generated by the LWM2M server to increase the response entropy.
    - *Argument 3:* Key Index: when needed, used to select the key which has to be used for the cryptographic operation

2. ***AuthData*:** defines a String Resource which receives the data generated according to the *GetAuthData* request. This resource is a Read-Only resource.

    - when *AuthBuffer* resource receives valid information, the *AuthStatus* MUST be turned to DATA_AVAIL_STATE

    - when *AuthBuffer* resource is accessed through a LWM2M READ request or a LWM2M NOTIFY, the *AuthStatus* MUST be reset to IDLE_STATE

3. ***AuthStatus*:** defines a Resource which contains the state related to the process triggered by *GetAuthData* request.

    0: IDLE_STATE : the buffer *AuthData* doesn't contain any valid data

    1: DATA_AVAIL_STATE : the buffer *AuthData* contains valid data

    2: ERROR_STATE : the process triggered by *GetAuthData* request incurred an error

    - the *AuthStatus* MUST be set to IDLE_STATE when a *GetAuthData* request is triggered
    - the *AuthStatus* MUST be set to DATA_AVAIL_STATE, when *AuthData is filled up with valid data*

- the *AuthStatus* MUST be set to ERROR_STATE when the process triggered by a *GetAuthData* request incurred an error

For supporting such Services, the Portfolio Objects resources of this sub-section MUST be bound to the interface of a Tamper-resistant component (e.g. embedded SE as defined in [GP]). This binding is implementation specific.

The Tamper-resistant component MUST be provisioned with a dedicated cryptographic key set for each Portfolio Object Instance Service. The cryptographic key sets MUST NOT be shared by the different Portfolio Object Instances.

5.2.2 Services Description

1. **Data Authentication:** this Service will be used when an *Identity* Resource Instance of a Portfolio Object Instance must be certainly authenticated and must be guaranteed against cloning threats.

   Strong authentication requires the cryptographic capability provided by a Tamper-resistant component which MUST be provisioned according to the security policy rules of the Application Provider.

   At least two data items MUST be provisioned in the Tamper-resistant component:

   - a Hash of the identity to authenticate
   - a cryptographic keyset shared with the Server intended to authenticate the identity above.

   Cryptographic Process: when a *GetAuthData* request is forwarded to the Tamper-resistant component, this cryptographic element will process the request to generate a Signature of the specified *Identity* Resource Instance value which MUST be of the following form:

   $$\textbf{Signature} = \textbf{SIG}_{\textbf{Kn}}\textbf{(Hash(}\textit{\textbf{Identity}}\textbf{)} + \text{Challenge}\textbf{)}$$

   Where:

   - **SIG** is a signature function shared between the Tamper-resistant component and the Server qualified to authenticate the Identity
   - **Kn** is one **cryptographic key of the** Data Authentication Service keyset provisioned in the Tamperresistant component and selected by the Argument 3 of the GetAuthData request.
   - **Hash(*Identity*)** is the data provisioned in the Tamper-resistant component and selected by the Argument 1 of the request
   - **Challenge** is a random value provided by the Server in Argument 2 of the GetAuthData request
   - **Signature** is the result of this cryptographic process which will be stored in the *AuthData* resource, and will be transported by the LWM2M protocol for authentication by the qualified Server.

   Note: the properties of the Hash and Signature functions as well as the cryptographic key set, only have to be known between the Tamper-resistant component and the Server intended to authenticate the provided Identity; that's why these elements are out of the scope of the Portfolio Object Specification (e.g. Hash function can be based on SHA-2 algorithm, and the Signature function can use the AES-CMAC algorithm specified in [RFC4493]).

2. **Data Encryption:** this Service will be used when a data contained in an *Identity* Resource Instance of a Portfolio Object Instance needs to be encrypted.

   At least one cryptographic key must be provisioned in the Tamper-resistant component

   Cryptographic Process: when a *GetAuthData* request is forwarded to the Tamper-resistant component, this cryptographic element will process the request to cipher the specified *Identity* Resource Instance value. The cipher function MUST be of the form:

   $$\textbf{Encryption} = \textbf{ENCRYPT}_{\textbf{Kn}}\textbf{(}\textit{\textbf{data}} + \text{Challenge}\textbf{)}$$

   Where:

   - **ENCRYPT** is a cryptographic function (e.g. AES-128 function) shared between the Tamper-resistant component and the Server intended to decrypt the data
   - **Kn** is one **cryptographic key of the** Data Encryption Service key set provioned in the Tamper-resistant component and selected by the Argument 3 of the GetAuthData request.

- **data** is the content of the Identity Resource Instance as selected by the Argument 1 of the GetAuthData request
- **Challenge** is a random value provided by the Server in Argument 3 of the GetAuthData request
- **Encryption** is the result of this cryptographic process which will be stored in the *AuthData* resource.

6. LWM2M Object: Portfolio

Description

This section formalizes the Resources definitions of the Portfolio Object described in section 5 of this document.

Object definition

Name	Object ID	Instances	Mandatory	Object URN
Portfolio	16	Multiple	Optional	urn:oma:lwm2m:oma:16

Resource definition

ID	Name	Operations	Instances	Mandatory	Type	Range or Enumeration	Units	Description
0	Identity	RW	Multiple	Mandatory	String			Data Storage extension for other Object Instances. e.g for [GSMA]: 0: Host Device ID, 1: Host Device Manufacturer 2: Host Device Model 3: Host Device Software Version, This Resource contains data that the *GetAuthData* executable Resource can work with.
1	GetAuthData	E	Single	Optional	none			Executable resource to trigger Services described in Section 5.2.2 Arguments definitions are described in Section 5.2.1 as well as in table 2 of this document
2	AuthData	R	Multiple	Optional	String			Buffer which contains the data generated by the process triggered by a *GetAuthData* request
3	AuthStatus	R	Single	Optional	Integer	[0-2]		This Resource contains the state related to the process triggered by *GetAuthData* request. 0: IDLE_STATE: *AuthData* doesn't contain any valid data 1: DATA_AVAIL_STATE: *AuthData* contains valid data 2: ERROR_STATE: an error occurred This state is reset to IDLE_STATE, when the executable resource "GetAuthData" is triggered or when the AuthData resource has been returned to the LWM2M Server (READ / NOTIFY).

Table 1 : Portfolio Object Resources Definition

Execution Resource Arguments definition

ID	Resource Name	Order	Name	Type	Range or Enum	Unit	Description
1	GetAuthData	0	Service	Integer	[0-1]		Data Security Services 0: SRV_AUTHENT: Authentication 1: SRV_ENCRYPT: Encryption

		1	Identity Instance Id	Integer	-	-	Identity Resource Instance ID of the current Portfolio Object Instance
		2	Challenge	Integer			A random provided by the Server
		3	Key Index	Integer			To select the cryptographic key in the Service dedicated key set provisionedin the Tamper-resistant component
		4..9	-	String			Application dependent arguments

Table 2: GetAuthData Executable Resource Arguments Definition

Appendix A. Change History (Informative)

A.1 Approved Version History

Reference	Date	Description
n/a	n/a	No prior version

A.2 Draft/Candidate Version 1.0 History

Document Identifier	Date	Sections	Description
Draft versions OMA-TS-LWM2M_PortfolioObj-V1_0	12 Dec 2015	all	First draft
	25 May 2016	all	Incorporated CRs: OMA-DM-LightweightM2M-2016-0030R05-CR_Porfolio_Basics OMA-DM-LightweightM2M-2016-0031R06-CR_Portfolio_Extended
	17 Aug 2016	B.2	Editorial changes
Candidate version OMA-TS-LWM2M_PortfolioObj-V1_0	06 Sep 2016	n/a	Status changed to Candidate by TP TP Ref # OMA-TP-2016-0092R01-INP_LWM2M_PortfolioObj_V1_0_RRP_for_Candidate_approval

Appendix B. Portefolio Object Usage Examples

B.1 Illustration I: Current GSMA CLP.03 requirement [GSMA]

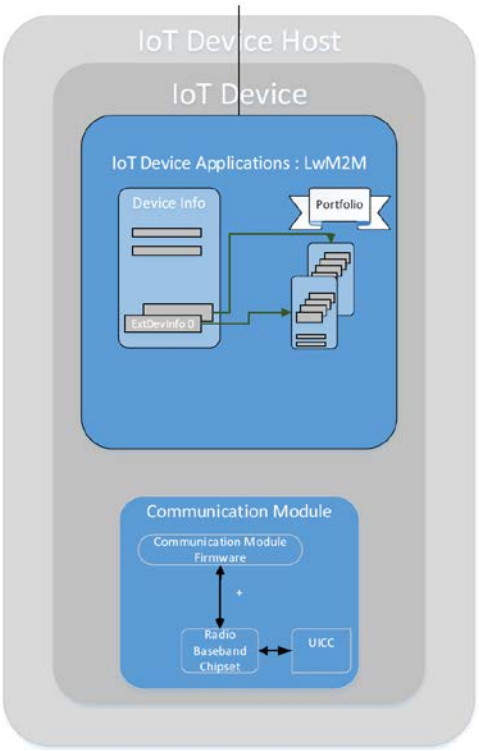

Figure 1: GSMA CLP.03 Requirement (IoT Device Connection Efficiency Guidelines)

In the GSMA CLP.03 document, there are specific requirements (DID4-7) related to the mandatory capability of the Device Management Technology. LWM2M TS 1.0 using Portfolio Object functionality is able to cover such a requirement as illustrated below:

DID4	*The following LWM2M resource has been defined to specify information related to the manufacturer of the IoT Host Device, this field will need to match the IoT Device Host manufacturer name that is referenced in the Mobile Network Operator lab certification of the IoT Device.* ***Type:*** *Host Device Manufacturer* ***Occurrence:*** *One* ***Format:*** *String* ***Name:*** */16/x/0/1 (HostMan)* ***Access Type****: READ* *The IoT Device Host manufacturer will be maintained in this resource by the Communications Module LWM2M client.*
DID5	*The following LWM2M resource has been defined to specify the Model name/number of the IoT Device Host. This shall match the model name/number used in the certification of the IoT Device.* ***Type:*** *Host Device Model* ***Occurrence:*** *One* ***Format:*** *String* ***Name:*** */16/x/0/2 (HostMod)* ***Access Type****: READ* *The IoT Host Device model will be maintained in the node by the Communication Module LWM2M client.*

DID6	*The following LWM2M resource has been defined to specify the software version of the IoT Device Host, this information shall be populated by the IoT Device Host manufacturer, shall match the version of SW certified by PTCRB and must be updated whenever the SW is updated on the device.* ***Type:*** *Host Device Software Version* ***Occurrence:*** *One* ***Format:*** *String* ***Name:*** */16/x/0/3 (HostSwV)* ***Access Type:*** *READ* *The IoT Host Device software version will be maintained in this resource by the Communication Module LWM2M client*
DID7	*The following OMA-DM node has been defined to specify the unique ID allocated to the IoT Device Host by the certifying Mobile Network Operator. Mobile Network Operators' may decide to include this field if they need a way to monitor for uncertified devices used on the network.* ***Type:*** *Host Device Unique ID* ***Occurrence:*** *One* ***Format:*** *Alphanumeric String* ***Name:*** */16/x/0/0 (HostUniqueID)* ***Access Type:*** *READ* *The IoT Device Host Unique ID is assigned by the Mobile Network Operator and will be stored in this resource.*

In a given LWM2M Client implementation, the graphical data representation of the GSMA-compatible Connection Module can be illustrated as follows (e.g. the GSMA Portfolio Object Instance ID is assigned to /16/1).

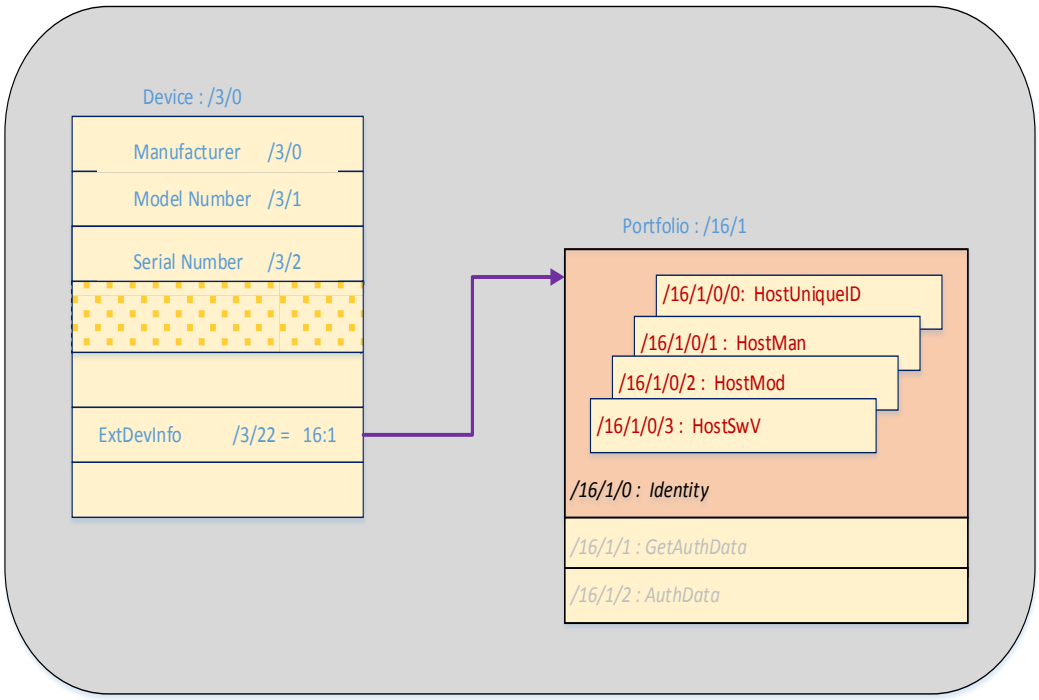

Figure 2: LWM2M Object Device (ID:3) extended in using a link to a dedicated GSMA Portfolio Object Instance

B.2 Illustration II: Extended usage

The use cases below provide only few examples on how the association of the LWM2M Client, the Portfolio Object and a Tamper-resistant component can work together. It is left to implementation to further enhance the capabilities of such an association.

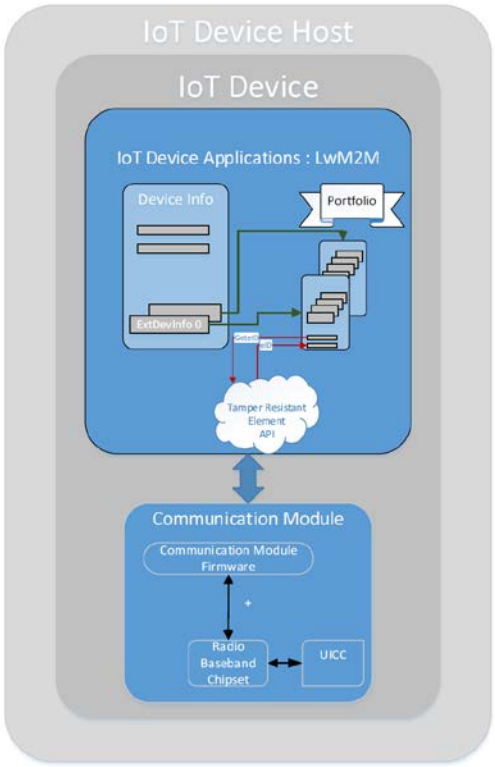

Figure 3: End-to-end security for application

Coupled with a Tamper-resistant component, the LWM2M Client and the Portfolio Object may support Authentication and end-to-end data encryption services exposed to applications.

Use Case 1: Vehicle (Host Device) Authentication

A Car Manufacturer Server decides to contact a vehicle (Host Device) and would like to rely on the Host Device ID managed by the LWM2M Client of the vehicle to authenticate that car.

The LWM2M Client and its association with a Tamper-resistant component have been configured according to an out-of-band process specific to the Car Manufacturer security policy. Typically, the Tamper-resistant component has to be provisioned with the Hash of the Host Device ID and a key shared with the LWM2M Server of the Car Manufacturer.

Following the LWM2M process with the GSMA-based Portfolio functionality **guaranties** that the Host Device ID is really unique on the network.

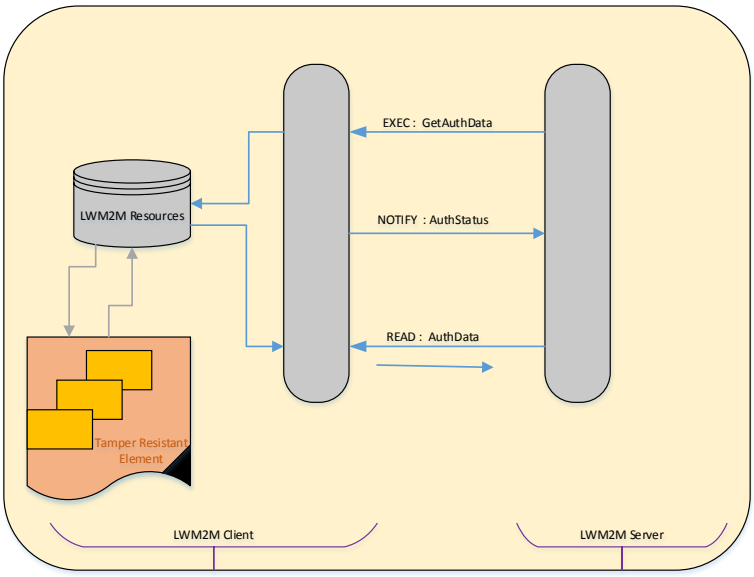

Figure 4: Process Illustration for Host Device Authentication

Host Device Authentication Process

Initial Step: Provisioning: in a secure environment according to the Car Manufacturer security policy, the Tamper-resistant component is provisioned with:

- a key- K_0 - shared with the Car Manufacturer LWM2M Server
- a hash of the Host Device ID: Hash(Host Device ID)

Step 1) Host Device Secured Identification Request

a) An initial request is initiated by the LWM2M Server in addressing the Executable resource *GetAuthData* of the Portfolio Object Instance with the proper parameters (the *DataStatus* resource is reset to IDLE_STATE):

e.g. LWM2M command: **EXECUTE /16/0/1 0=SRV_AUTHENT, 1=3, 2=0x2345178, *3=0***

- argument 0: An Authentication process is requested
- argument 1: the Identity Resource Instance 3 is the target (e.g. Unique Host Device ID)
- argument 2: 0x2345178 : a Challenge
- argument 3: the cryptographic key 0 of the Authentication key set must be used

b) Information is exchanged between the Tamper-resistant component and the LWM2M Client:

In the Tamper Resistant Component a signature is generated in using the pre-registered key 0 and the hash of the Host Device ID.The challenge is used for introducing a random in the signature.

Step 2) the LWM2M Server is polling the *AuthStatus* Resource or is waiting for a notification related to the availability of the *AuthData* Resource

Step 3) when available, a Signature of the requested *Identity* Resource Instance [Sig (Hash(Host Device ID)+ Challenge)] is delivered by the Tamper-resistant component into the *AuthData* Resource of the Portfolio Object Instance.

Step 4) when the Signature is available in the Portfolio Instance (DataStatus Resource is set to DATA_AVAIL_STATE), either a notification is sent to the LWM2M Server with such data or the LWM2M Server will perform a READ request on the *AuthData* resource(s) of the Portfolio Instance. In both cases the *DataStatus* Resource is reset to IDLE-STATE.

When receiving this Signature, the LWM2M Server which initiated the request can authenticate the identity of the vehicle with the highest degree of confidence.

Use Case 2: Encryption of sensitive data from the Portfolio Object Instance

Very similar to the "Use Case 1" process, this new use case is addressing the need to answer to the Server request for returning sensitive data in encrypted form. The initial data is present in a Portfolio Object Instance (i.e. in one Instance of the *Identity* multi-instance Resource).

LWM2M Server request: EXECUTE /16/0/GetAuthData 0=SEC_ENCRYPT, 1=2, 2=0x2345178, *3=1*

- argument 0: the data encryption service is requested (SEC_ENCRYPT)
- argument 1: the Identity Instance of ID=2 is the target (e.g. Host Device Model)
- argument 2: 0x2345178: the Challenge
- argument 3: the cryptographic key 1 of the Authentication key set must be used.

When generated by the Tamper-resistant component, the *AuthData* Resource will contain an encrypted value for e.g. the Host Device Model data