ETSI TS 118 103 V4.7.1 (2026-03)

TECHNICAL SPECIFICATION

oneM2M; Security Solutions
(oneM2M TS-0003 version 4.7.1 Release 4)

Reference
DTS/oneM2M-0003v4

Keywords
IoT, M2M, SECURITY

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI IPR online database.

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™**, **LTE™** and **5G™** logo are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM®** and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Technical Specification (TS) has been produced by ETSI Partnership Project oneM2M (oneM2M).

Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document defines security solutions applicable within the M2M system.

2 References

2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found in the ETSI docbox.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are necessary for the application of the present document.

[1] ETSI TS 118 101: "oneM2M; Functional Architecture (oneM2M TS-0001)".

[2] ETSI TS 118 111: "oneM2M; Common Terminology (oneM2M TS-0011)".

[3] Void.

[4] ETSI TS 118 104: "oneM2M; Service Layer Core Protocol Specification (oneM2M TS-0004)".

[5] IETF RFC 5246: "The Transport Layer Security (TLS) Protocol Version 1.2".

NOTE 1: TLS 1.3 will be considered in a future release of the document.

NOTE 2: Obsoleted by IETF RFC 8446.

[6] IETF RFC 6347: "Datagram Transport Layer Security Version 1.2".

NOTE: Obsoleted by IETF RFC 4347.

[7] ETSI TS 102 225 (V11.0.0): "Smart Cards; Secured packet structure for UICC based applications (Release 11)".

[8] ETSI TS 102 226 (V11.0.0): "Smart Cards; Remote APDU structure for UICC based applications (Release 11)".

[9] ETSI TS 131 115 (V10.1.1): "Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); Secured packet structure for (Universal) Subscriber Identity Module (U)SIM Toolkit applications (3GPP TS 31.115 version 10.1.1 Release 10)".

[10] ETSI TS 131 116 (V10.2.0): "Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE; Remote APDU Structure for (U)SIM Toolkit applications (3GPP TS 31.116 version 10.2.0 Release 10)".

[11] TIA TIA-1106: "Secured Packet Structure for CCAT Applications".

[12] TIA TIA-1107: "Remote APDU Structure for CCAT Applications".

[13] ETSI TS 133 220: "Digital cellular telecommunications system (Phase 2+) (GSM); Universal Mobile Telecommunications System (UMTS); LTE; Generic Authentication Architecture (GAA); Generic Bootstrapping Architecture (GBA) (3GPP TS 33.220)".

[14] TIA TIA-1098: "Generic Bootstrapping Architecture (GBA) Framework".

[15] IETF RFC 4279: "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)".

[16] Void.

[17] Void.

[18] IETF RFC 5705: "Keying Material Exporters for Transport Layer Security (TLS)".

[19] IETF RFC 3629: "UTF-8, a transformation format of ISO 10646".

[20] Unicode 13.0.0 UAX #15: "Unicode® Standard Annex #15; Unicode Normalization Forms", February 2020.

[21] GlobalPlatform® GPD_SPE_120: "TEE Management Framework v1.0".

[22] GlobalPlatform® GPD_SPE_009: "TEE System Architecture v1.1".

[23] ETSI TS 102 671: "Smart Cards; Machine to Machine UICC; Physical and logical characteristics".

[24] ETSI TS 102 221: "Smart Cards; UICC-Terminal interface; Physical and logical characteristics".

[25] ETSI TS 102 484: "Smart Cards; Secure channel between a UICC and an end-point terminal".

[26] ISO/IEC 7816-4: "Identification cards -- Integrated circuit cards -- Part 4: Organization, security and commands for interchange".

[27] ETSI TS 101 220: "Smart Cards; ETSI numbering system for telecommunication application providers".

[28] Void.

[29] Void.

[30] Void.

[31] IETF RFC 6655: "AES-CCM Cipher Suites for Transport Layer Security (TLS)".

[32] IETF RFC 5289: "TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)".

[33] IETF RFC 2104: "HMAC: Keyed-Hashing for Message Authentication".

[34] IETF RFC 5280: "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile".

[35] IETF RFC 6960: "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP".

[36] IETF RFC 6961: "The Transport Layer Security (TLS) Multiple Certificate Status Request Extension".

[37] IETF RFC 7250: "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)".

[38] Void.

[39] NIST Federal Information Processing Standard (FIPS) 186-4: "Digital Signature Standard (DSS)".

[40] IETF RFC 6920: "Naming Things with Hashes".

[41] IETF RFC 4648: "The Base16, Base32, and Base64 Data Encodings".

[42] IETF RFC 5487: "Pre-Shared Key Cipher Suites for TLS with SHA-256/384 and AES Galois Counter Mode".

[43] IETF RFC 8422: "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) Versions 1.2 and Earlier".

[44] IETF RFC 6066: "Transport Layer Security (TLS) Extensions: Extension Definitions".

[45] IETF RFC 7251: "AES-CCM Elliptic Curve Cryptography (ECC) Cipher Suites for TLS".

[46] IETF RFC 5480: "Elliptic Curve Cryptography Subject Public Key Information".

[47] GlobalPlatform® GPD_SPE_008: "Secure Element Remote Application Management v1.0.1".

[48] IETF RFC 5869: "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)".

[49] IETF RFC 7518 (2015): "JSON Web Algorithms (JWA)".

[50] IETF RFC 7516 (2015): "JSON Web Encryption (JWE)".

[51] IETF RFC 7515 (2015): "JSON Web Signature (JWS)".

[52] W3C® Recommendation: "XML Signature Syntax and Processing v1.1", 2013.

[53] IETF RFC 7519 (2015): "JSON Web Token (JWT)".

[54] OpenID Foundation: "OpenID Connect Core 1.0", 2014.

[55] W3C® Recommendation: "XML Encryption Syntax and Processing v1.1", 2013.

[56] Void.

[57] ETSI TS 118 122: "oneM2M; Field Device Configuration (oneM2M TS-0022)".

[58] ETSI TS 118 132: "MAF and MEF Interface Specification (oneM2M TS-0032)".

[59] IETF RFC 7030: "Enrollment over Secure Transport".

[60] Void.

[61] Void.

[62] Void.

[63] GlobalPlatform® GPC_SPE_034: "Card Specification v2.3.1".

[64] CEN EN 419 212-1:2017: "Application Interface for Secure Elements For Electronic Identification, Authentication and Trusted Services – Part 1: Introduction and common definitions".

[65] Void.

[66] IETF RFC 8894: "Simple Certificate Enrolment Protocol".

[67] ETSI TS 118 116: "oneM2M; Secure Environment Abstraction (oneM2M TS-0016)".

[68] Void.

[69] NIST Federal Information Processing Standard (FIPS) 201-2: "Personal Identity Verification (PIV) of Federal Employees and Contractors".

NOTE: Superseded by FIPS 201-3.

[70] GSMA: "SGP.01 - Embedded SIM Remote Provisioning Architecture".

[71] NIST Federal Information Processing Standard (FIPS) 186-2: "Digital Signature Standard (DSS)".

[72] IETF RFC 5116 (2008): "An interface and algorithms for authenticated Encryption".

[73] ISO 9797 (2011): "Information Technology - Security Techniques -- Message Authentication Codes (MACs)".

[74] SOG-IS: "SOG-IS Crypto Evaluation Scheme Agreed Cryptographic Mechanisms", Version 1.2, January 2020.

[75] IETF RFC 5639: "Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation".

[76] ETSI TS 118 108: "oneM2M; CoAP Protocol Binding (oneM2M TS-0008)".

[77] ETSI TS 123 246 (V15.0.0): "Universal Mobile Telecommunications System (UMTS); LTE; Multimedia Broadcast/Multicast Service (MBMS); Architecture and functional description (3GPP TS 23.246 version 15.0.0 Release 15)".

[78] ETSI TS 133 246 (V14.2.0): "Universal Mobile Telecommunications System (UMTS); LTE; 3G Security; Security of Multimedia Broadcast/Multicast Service (MBMS) (3GPP TS 33.246 version 14.2.0 Release 14)".

[79] ISO 8601: "Date and time -- Representations for information interchange ".

[80] IETF RFC 7525: "Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)".

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents may be useful in implementing an ETSI deliverable or add to the reader's understanding, but are not required for conformance to the present document.

[i.1] oneM2M Drafting Rules.

[i.2] Void.

[i.3] Void.

[i.4] ETSI TR 118 508: "Analysis of Security Solutions for the oneM2M System (oneM2M TR-0008)".

[i.5] eXtensible Access Control Markup Language (XACML) Version 3.0. 22 January 2013. OASIS Standard.

[i.6] Handbook of Applied Cryptography, A. J. Menezes, P. C. van Oorschot, S. A. Vanstone, CRC Press, 1996.

[i.7] Recommendation ITU-T X.509 (10/2016): "Information technology - Open Systems Interconnection - The Directory: Public-key and attribute certificate frameworks".

[i.8] Void.

[i.9] OMA-TS-REST-NetAPI-TerminalLocation-V1-0-20130924-A: "RESTful Network API for Terminal Location", Version 1.0.

[i.10] ISO 3166-1:2013: "Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes".

[i.11] ISO/IEC 7816-5: "Identification cards - Integrated circuit cards - Part 5: Registration of Application Providers".

[i.12] NIST Special Publication 800-162: "Guide to Attribute Based Access Control (ABAC) Definition and Considerations".

[i.13] National Institute of Standards and Technology: "Guide to Protecting the Confidentiality of Personally Identifiable Information (PII)".

[i.14] Void.

[i.15] oneM2M TR-0019: "Dynamic Authorization for IoT".

[i.16] ETSI TR 118 512: "oneM2M; End-to-End security and Group Authentication (oneM2M TR-0012)".

[i.17] Void.

[i.18] IANA JSON Web Token (JWT) registry.

[i.19] IETF RFC 6455: "The Web Socket Protocol", December 2011.

[i.20] IETF RFC 7230: "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing".

[i.21] IETF RFC 7252: "The Constrained Application Protocol (CoAP)".

[i.22] Void.

[i.23] Void.

[i.24] Void.

[i.25] Void.

[i.26] https://github.com/certnanny/sscep.

[i.27] https://github.com/jscep/jscep.

[i.28] https://github.com/certnanny/sscep/issues/42.

[i.29] ETSI TS 118 105: "oneM2M; Management Enablement (OMA) (oneM2M TS-0005)".

[i.30] ETSI TS 118 106: "oneM2M; Management Enablement (BBF) (oneM2M TS-0006)".

[i.31] Broadband Forum TR-069: "CPE WAN Management Protocol".

[i.32] BSI TR 03109: "Smart Meter Gateway specification".

[i.33] ETSI TS 103 645: "CYBER; Cyber Security for Consumer Internet of Things: Baseline Requirements".

[i.34] IoTF Security Foundation. Secure Design Best Practice Guide, Release 2 November 2019.

[i.35] GSMA - IoT Security Guidelines and Assessment.

3 Definition of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in ETSI TS 118 111 [2] and the following apply:

additional authenticated data [14]: data that is authenticated, but not encrypted by an authenticated encryption with associated data algorithm

AE-ID Certificate: certificate with a certificate chain to a trust anchor certificate and containing an AE-ID in the subjectAltName extension

NOTE: An AE_ID certificate can be used to verify that an entity has been assigned the AE-ID in the certificate.

association configuration: phase of a Security Association Establishment Framework in which the entity establishing the Security Association (and the Central Key Distribution Server, in the case of Centralized Security Frameworks), are provided with identities (and any other relevant credentials) to ensure that the security association is established between the intended entities

association security handshake: phase of a Security Association Establishment Framework in which the security association endpoints perform mutual authentication

authenticated encryption with associated data [14]: algorithm providing confidentiality for the plaintext and a way to check its integrity and authenticity while providing the ability to check the integrity and authenticity of some additional authenticated data

NOTE: In this context plaintext refers to data that is authenticated and encrypted.

bootstrap credential: pre-provisioned credential enabling mutual authentication of the Enrolee and the M2M Enrolment function

bootstrap credential configuration: phase of a Remote Security Provisioning Framework in which the Bootstrap Credentials are pre-provisioned to the Enrolee and the M2M Enrolment function

bootstrap enrolment handshake: phase of a Remote Security Provisioning Framework in which the Enrolee and M2M Enrolment Function perform mutual authentication

bootstrap instruction configuration: phase of a Remote Security Provisioning Framework in which the Enrolee and M2M Enrolment Function are provided with identities (and any other relevant credentials) to enable the M2M Enrolment function to establish a Master Credential between the intended Enrolee and M2M Authentication Function

bootstrap server function [13]: BSF is hosted in a network element under the control of a Mobile Network Operator. BSF, HSS, and UEs participate in GBA in which a shared secret is established between the network and a UE by running the bootstrapping procedure

NOTE: The shared secret can be used between NAFs and UEs, for example, for authentication purposes.

bootstrapping transaction identifier [13]: bootstrapping transaction identifier (B-TID) is used to bind the subscriber identity to the keying material in GBA reference points Ua, Ub and Zn

CA-Certificate [i.6]: certificate created by one Certification Authority (CA) certifying the public key of another CA

certificate: See Public Key Certificate.

certificate authority: trusted entity that issues digital certificates

certificate chain: sequence of one or more CA-certificates, where: the Public Verification Key in each CA-certificate is certified in the previous CA-certificate; and the public key of the first CA-Certificate is trusted *a priori*

NOTE: Trust in the public key in each CA-certificate can be based on trust in the previous CA-Certificate.

certificate name: unique identifier in a name field of a Certificate (e.g. in the X.509 "Subject" or "Subject Alternative Name" attribute)

certificate provisioning (procedure): procedure performed by a Security Principal and a MEF for provisioning the Security Principal with an MEF-Provisioned Certificate and Certificate(s) of the MEF Certificate Authority

NOTE: Additional Certificate Authority Certificates can also be provisioned via other means such as pre-provisioning or ETSI TS 118 122 [57].

certificate re-provisioning (procedure): Certificate Provisioning procedure performed when the Security Principal can authenticate itself with a valid Enrolled Certificate

certificate signing request: message used to request a Public Key Certificate

certificate verification: process necessary to trust an entity's Certificate

certification authority [i.6]: authority responsible for establishing and vouching for the authenticity of public keys

NOTE: This includes binding public keys to distinguished names through signed certificates, managing certificate serial numbers, and certificate revocation.

content encryption key: symmetric key used to encrypt plaintext to produce the ciphertext and generate a Message Integrity Check (MIC)

NOTE: In Authenticated Encryption with Associated Data (AEAD), the content encryption key is used directly, while in other algorithms the content encryption key is used to generate distinct keys for the encryption algorithm and integrity protection algorithm.

credential configuration: phase of a Security Association Establishment Framework in which the Credentials necessary for the Security Association Establishment Framework are configured to the relevant entities and functions

Credential-ID type-ID: portion of a Credential-ID indicating the type of credential being identified

CSE-ID certificate: certificate with a certificate chain to a root of trust and containing a CSE-ID in the subjectAltName extension

NOTE: A CSE_ID certificate can be used to verify that an entity has been assigned the CSE-ID in the certificate.

device certificate: certificate with a certificate chain to a root of trust and containing at least one globally unique hardware instance identifier in the subjectAltName extension

NOTE: A device certificate can be used to verify that an entity is executing on the identified hardware instance.

digital signature [i.7]: information is signed by appending to it an enciphered summary of the information

NOTE: The summary is produced by means of a one-way hash function, while the enciphering is carried out using the private key of the signer.

ESCertKE Initiating End-Point: ESCertKE end-point which initiates the ESCertKE Procedure

ESCertKE Messages: messages exchanged between the ESCertKE Initiating End-Point and ESCertKE Terminating End-Point as part of End-to-End Certificate-based Key Establishment

ESCertKE Procedure: sequence of exchanged ESCertKE Messages and processing based on those ESCertKE Messages, for the purposes of End-to-End Certificate-based Key Establishment

ESCertKE Terminating End-Point: ESCertKE end-point with which the ESCertKE Initiating End-Point intends to perform the ESCertKE Procedure

End-to-End Certificate-based Key Establishment: interoperable framework for two end-points to use certificates for establishing end-to-end secret symmetric keys for use in other end-to-end security frameworks such as End-to-End Security of Data (ESData) or End-to-End Security of Primitives (ESPrim)

End-to-End security of data: interoperable framework for protecting data that ends up transported using oneM2M reference points, in order that transited CSEs do not need to be trusted with that data

End-to-End Security of Primitives: interoperable framework for securing the exchange of oneM2M primitives so that CSEs do not need to be trusted with the confidentiality and integrity of the primitives

enrolee: AE or CSE that requires remote provisioning of a symmetric key to be shared with an enrolment target

enrolment key: symmetric key established between an Enrolee and M2M Enrolment Function following successful mutual authentication

NOTE: A symmetric key to be shared by the Enrolee and an Enrolment Target may be derived (at the Enrolee and M2M Enrolment Function) from the currently valid Enrolment Key, and the M2M Enrolment Function subsequently securely delivers the symmetric key to the Enrolment Target.

enrolment key generation: phase of remote security provisioning Framework in which the Enrolee and M2M Enrolment function establish an Enrolment Key and Enrolment Key identifier

enrolment phase: step in the lifecycle of an M2M equipment where it becomes provisioned for operation with a specific M2M Service Provider

enrolment target: M2M Authentication Function, CSE, or AE with whom an Enrolee wishes to establish a symmetric key (master credential or pre-provisioned secure connection key) using remote security provisioning

entity identifier: CSE-ID (or AE-ID respectively) of a CSE (or AE respectively)

ESData Envelope: data object containing the result of protecting an End-to-End Security of Data (ESData) Payload using the ESData procedures

ESData Payload: data to be protected using End-to-End Security of Data (ESData)

FQDN certificate: certificate with a certificate chain to a root of trust and containing an FQDN

generic bootstrap architecture: set of 3GPP and 3GPP2 specifications providing security features and a mechanism to bootstrap authentication and key agreement for application security from the 3GPP and 3GPP2 underlying network authentication mechanisms

initial certificate provisioning (procedure): Certificate Provisioning procedure performed when the Security Principal cannot authenticate itself with a valid Enrolled Certificate

inner request primitive: request primitive to be protected by End-to-End Security of Primitives (ESPrim)

inner response primitive: response primitive to be protected by End-to-End Security of Primitives (ESPrim)

message integrity code: tag computed from a message and a symmetric key, and attached to a message

NOTE 1: The purpose of a messages integrity code is to facilitate, without the use of any additional mechanisms, assurances regarding both the source of a message and its integrity.

NOTE 2: A Message Integrity Code is sometimes called a "Message Authentication Code" - "Message Integrity Code" has been used since the abbreviation of "Message Authentication Code" (MAC) might be misunderstood to refer to "Media Access Control". The definition is based on text from [i.6] (p323).

M2M secure connection key: symmetric key established between two entities (CSEs or AEs), by an of M2M Authentication Function, in order to secure the communication between those two entities

NOTE: This M2M Secure Connection Key results from a successful M2M Security Association Establishment procedure.

M2M trust enabler: stakeholder trusted with enabling authentication of CSEs/AEs to other CSEs/AEs

MAF client: CSE or AE configured to use the services of an M2M Authentication Function

MAF Credential Configuration procedure: MAF Security Framework procedure used for Enrolment Phase of an End-Point by establishing credentials for mutual authentication between an End-Point and an MAF

MAF handshake procedure: MAF Security Framework procedure in which an entity and the MAF perform mutual authentication and generate a Symmetric Key which can then be used in the Association Security Handshake for mutual authentication between that entity and other entities

MAF key registration procedure: MAF Security Framework procedure in which a Source End-Point and the MAF generate a Symmetric Key which can then be used for mutual authentication between the Source End-Point and one or more other Target End-Points

MAF key retrieval procedure: MAF Security Framework procedure in which a Target End-Point retrieves the Symmetric Key previously generated by the MAF and a Source End-Point, to enable mutual authentication between the Source End-Point and the Target End-Point

master credentials: credentials used to mutually authenticate between an ASN/MN-CSE and the MAF. This is done to secure access to the infrastructure of an M2M Service Provider

NOTE: The Master Credentials are either pre-provisioned or remotely provisioned (without relying on those credentials).

MEF certificate authority: role of a Certificate Authority which issues MEF-Provisioned Certificates to a Security Principal through the MEF

NOTE: The term is relative to the MEF, so an MEF Certificate Authority with respect to one MEF is not an MEF Certificate Authority with respect to another MEF

MEF client: functionality for performing MEF procedures on behalf of an associated CSE or AE, or on behalf of CSE or AE(s) present on an associated Node, or an associated MAF

MEF-provisioned certificate: certificate issued by a Certificate Authority, via an MEF, for authenticating the Security Principal

NOTE: The term is relative to the MEF, so a MEF-Provisioned Certificate with respect to one MEF is not a MEF-Provisioned Certificate with respect to another MEF.

Node-ID Certificate: certificate with a certificate chain to a root of trust and containing a M2M-Node-ID of a Node in the subjectAltName extension

NOTE: A Node-ID certificate can be used to verify the identity of a Node.

(oneM2M) security principal: CSE or AE or Node or M2M Device which can be authenticated

NOTE: When the Security Principal is a Node or M2M Device, then Node or M2M Device is acting on behalf of the CSE and/or AE executing on the Node or M2M Device.

Online Certificate Status Protocol: protocol for requesting a report on the status of one or more X.509 certificates (IETF RFC 6960 [35])

operational phase: period in the lifecycle of an M2M equipment where it is actually used for providing M2M services

outer request Primitive: request primitive used to transport the data object containing an inner request primitive to which End-to-End Security of Primitives (ESPrim) has been applied

outer response Primitive: response primitive used to transport the data object containing an inner response primitive to which End-to-End Security of Primitives (ESPrim) has been applied

Personally Identifiable Information [i.13]: any information about an individual maintained by an agency, including:

1) any information that can be used to distinguish or trace an individual's identity, such as name, social security number, date and place of birth, mother's maiden name, or biometric records; and

2) any other information that is linked or linkable to an individual, such as medical, educational, financial, and employment information.

policy decision point [i.5]: system entity that evaluates applicable policy and renders an authorization decision

policy enforcement point [i.5]: system entity that performs access control, by making decision requests and enforcing authorization decisions

policy information point [i.5]: system entity that acts as a source of attribute values

policy retrieval point: system entity that retrieves applicable policy or policy set

pre-provisioned secure connection key: Symmetric Key that is pre-provisioned to two entities (which may be AEs or CSEs) to be used for mutual authentication of those entities in Security Association Establishment

pre-provisioned secure connection key identifier: Identifier for a Pre-Provisioned Secure Connection Key

pre-provisioned symmetric enrolee key: Symmetric Key that is pre-provisioned to the Enrolee and M2M Enrolment Function

pre-provisioned symmetric enrolee key identifier: identifier for a Pre-Provisioned Symmetric Enrolee Key

private signing key: secret key that can generate signatures that can be verified using a corresponding Public Verification Key

public key certificate: electronic document that uses a digital signature to bind a public key with an identity

NOTE: [i.6] A *public-key certificate* is a data structure consisting of a data part and a signature part. The data part contains cleartext data including, as a minimum, a public [verification] key and a string identifying the part (subject entity) to be associated therewith. The signature part consists of the digital signature of a certification authority over the data part, thereby binding the subject entity's identity to the specified public key.

public key certificate flavour: name describing the usage of a public key certificate within the scope of oneM2M

public key infrastructure: set of hardware, software, people, policies, and procedures needed to create, manage, distribute, use, store, and revoke Public Key Certificates

NOTE: For more details, see [i.6].

public verification key: credential that can verify digital signatures generated by a corresponding Private Signing Key, but which cannot be used to generate digital signatures

raw public key certificate: certificate comprising only the SubjectPublicKeyInfo structure of an X.509 certificate that carries the parameters necessary to describe the public key [37]

registration authority: functional entity responsible for verifying Certificate Signing Requests and authorizing a Certification Authority to issue a corresponding Certificate

relative enrolment key identifier: part of the enrolment key identifier that is unique within the context of a M2M Enrolment Function

secure environment: logical entity that protects Sensitive Data and Sensitive Functions from tampering, unauthorized monitoring or execution and that provides access to these Sensitive Data and Sensitive Functions to authorized oneM2M entities

security association establishment: sequential processing of credential configuration, association configuration and association security handshake between two entities

security association establishment framework: security framework for security association establishment

security bootstrap framework or **remote security provisioning framework:** mechanism for remotely provisioning a Master Credential and Master Credential Identifier to an Enrolee and an M2M Authentication Function

security framework: set of procedures providing Security Association Establishment or Remote security provisioning

security usage identifier: identifies a security feature (e.g. Security Association Establishment Framework, End-to-End Security of Primitives or End-to-End Security of Data), a protocol used for that security feature, and (where applicable) an option within a single protocol

NOTE: The security usage identifier is used to limit how a credential may be used.

self-signed certificate: Public Key Certificate that is signed by the same entity whose identity it certifies

sensitive function: function processed within the secure environment requiring protection from unauthorized monitoring, tampering or execution and that is operating on sensitive data, e.g. derivation of keys from M2M long-term service-layer keys and cryptographic algorithms

source ESData end-point: entity producing an End-to-End Security of Data (ESData) Envelope from an ESData Payload

symmetric key: secret key that is shared between two entities

target ESData end-point: entity producing the verified End-to-End Security of Data (ESData) Payload from an ESData Envelope

trust anchor certificate: certificate that is trusted a priori

X.509: Recommendation ITU-T for a Public Key Infrastructure

3.2 Symbols

For the purposes of the present document, the following symbols apply:

|| Concatenation

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in ETSI TS 118 111 [2] and the following apply:

(D)TLS-PSK (D)TLS Pre-Shared Key (ciphersuites)
3GPP2 3rd Generation Partnership Project 2

AAA	Authentication, Authorization and Accounting
ABAC	Attribute Based Access Control
ACP	Access Control Policy Instance
AEAD	Authenticated Encryption with Associated Data
AE-ID	Application Entity IDentifier
App-ID	Application IDentifier
ASE	Asymmetric Secure Element
ASN-CSE	Common Service Entity which resides in the Application Service Node
AuthorSignReqInfo	Authorization Signature Request Information
AuthorSign	Authorization Signature
AuthorRelMapRecord	Authorization Relationship Mapping Record
AuthorRelIndicator	Authorization Relationship Indicator
AuthorSignIndicator	Authorization Signature Indicator
BSF	Bootstrapping Server Function
B-TID	Bootstrapping Transaction IDentifier
CA	Certification Authority or Certificate Authority
CIDR	Classless Inter-Domain Routing
CoAP	Constrained Application Protocol
CSE-ID	Common Service Entity IDentifier
CSR	Certificate Signing Request
DTLS	Datagram Transport Layer Security (Protocol)
ECC	Elliptic Curve Cryptography
EKU	Extended Key Usage
ESCertKE	End-to-End Certificate-based Key Establishment
Enrolee-ID	Enrolee Identity
ESData	End-to-End Security of Data
ESF	End-to-End Security Function
ESPrim	End-to-end Security of Primitives
EST	Enrolment over Secure Transport
ETSI	European Telecommunications Standards Institute
FQDN	Fully Qualified Domain Name
GBA_ME	Mobile Equipment -based GBA
GBA_U	GBA with UICC-based enhancements
GUSS	GBA User Security Settings
HLR	Home Location Register
HSS	Home Subscriber System
HTTP	HyperText Transfer Protocol
HW	Hardware
ID	Identifier
IdA	Identifier for entity A
IdB	Identifier for entity B
IN-CSE	CSE which resides in the Infrastructure Node
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
IV	Initialization Vector
M2M-SP	Machine-toMachine Service Provider
MAF	M2M Authentication Function
MAF-ID	M2M Authentication Function Identifier
Mca	reference Point for M2M Communication with AE
Mcc	reference Point for M2M Communication with CSE
Mcc'	reference Point for M2M Communication with CSE of different M2M Service Provider
Mcn	reference Point for M2M Communication with NSE
MEF	M2M Enrolment Function
MIC	Message Integrity Code
MN-CSE	CSE which resides in the Middle Node
MTE	M2M Trust Enabler
NAF	Network Application Function
Node-ID	Node Identifier
OAEP	Optimal Asymmetric Encryption Padding
OCSP	Online Certificate Status Protocol
PDP	Policy Decision Point
PEP	Policy Enforcement Point

PII	Personally Identifiable Information
PIN	Personal Identification Number
PIP	Policy Information Point
PKI	Public Key Infrastructure
PRP	Policy Retrieval Point
RA	Registration Authority
RSA	Rivest, Shamir und Adleman
RSAES	RSA Encryption Scheme
RSASSA	RSA Signature Scheme Algorithm
RSPF	Remote Security Provisioning Framework
SAEF	Security Association Establishment Framework
SCEP	Simple Certificate Enrolment Protocol
SE	Secure Environment
SUID	Security Usage IDentifier
SW	SoftWare
T&C	Terms and Conditions
TEE	Trusted Execution Environment
TEF	Trust Enabling Function
TLS	Transport Layer Security (Protocol)
UE	(3GPP) User Equipment
UNSP	Underlying Network Service Provider
URI	Uniform Resource Identifier
USS	User Security Settings
XACML	eXtensible Access Control Markup Language

4 Conventions

The keywords "Shall", "Shall not", "May", "Need not", "Should", "Should not" in the present document are to be interpreted as described in the oneM2M Drafting Rules [i.1].

5 Security Architecture

5.1 Overview

5.1.0 Introduction

Figure 5.1.0-1 provides a high level overview of the Security architecture.

The architecture consists of following layers:

- Security Functions layer:
  - This layer contains a set of security functions that are exposed at reference point Mca and Mcc. These security functions can be classified into six categories; they are Identification, Authentication, Authorization, Security Association, Sensitive Data Handling and Security Administration.
- Security Environment Abstraction Layer:
  - This layer implements various security capabilities such as key derivation, data encryption/decryption, signature generation/verification, security credential read/write from/to the Secure Environments, and so on. The security functions in the Security Functions Layer invoke these functions in order to protect the operations in the Secure Environments. In addition this layer also provides physical access to the Secure Environments. Implementation of this is out of scope of the present document. This layer is specified in ETSI TS 118 116 [67].

- Secure Environment layer:

  - This layer contains one or multiple secure environments that provide various security services providing adequate protection to sensitive data storage and sensitive function execution. The sensitive data includes SE capability, security keys such as long term symmetric keys and asymmetric private keys, local credentials, security policies, identity information, subscription information, and so on. The sensitive functions include data encryption, data decryption, and so on. Though implementation of secure environments is out of scope of the present document, reference frameworks to interface M2M entities with common tamper-resistant hardware SE are provided in annexes D and L.

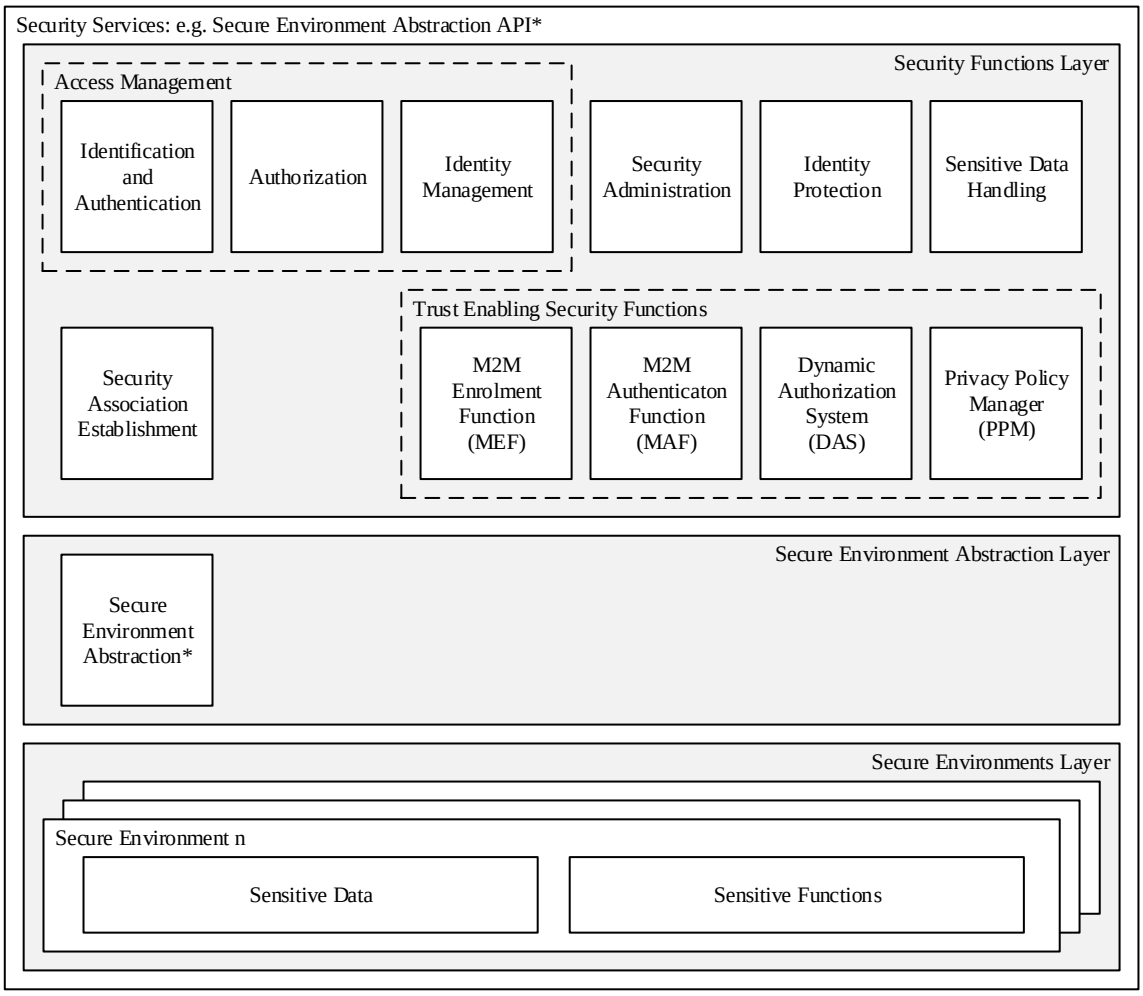


Figure 5.1.0-1: High level overview of the Security architecture

Design principles:

- Security Services are modular and configurable according to the needs of the hosting CSE, its supported reference points and its purpose.
- The architecture is split into several components and sub-components providing a modular design. With this design, mapping of the architecture to different nodes and entities is enabled.
- Depending on the requirements of each entity, Security consists of components relevant to fulfil the requirements of the respective node or entity and the intended use case.
- The architecture needs to be adapted to be suitable for implementation in different entities. For example, the architecture can be mapped to different device classes.
- The security administration component is supposed to enable administration of all sensitive resources (data and functions) and also allow configuration and extension of Security services itself.

- The Secure Environment within the CSE is accessed via the Secure Environment Abstraction layer and is expected to provide adequate level of protection to the sensitive information listed in clause 6.2.3.2.

5.1.1 Identification and Authentication

The Identification and Authentication function is in charge of identification and mutual authentication of CSEs and AEs.

Identification is the process of checking if the identity provided for authentication is valid. How to perform an identification process will depend on the purpose of authentication. For example, in the case of resource access, the authentication function can require the identification to check if the AE or CSE has registered with the local CSE; in the case of AE or CSE registration, the authentication function can require the identification to check if the identity provided by an AE or CSE fits a certificate from a trusted Certificate Authority. Once passing this checking process, the AE or CSE is identified, and the identified identity will be supplied to authentication process.

Authentication is the process of validating if the identity supplied in the identification step is associated with a trustworthy credential. How to perform an authentication process will depend on which mutual authentication mechanism is used. For example, in the case of using certificate based authentication mechanism, the authentication function can require the authentication to verify a digital signature; in the case of using symmetric key based authentication mechanism, the authentication function can require the authentication to verify a Message Integrity Code (MIC). When this validating process has been completed, the AE or CSE is authenticated.

5.1.2 Authorization

The Authorization function is responsible for authorizing services and data access to authenticated entities according to provisioned Access Control Policies (ACPs) and assigned roles.

Access control policy is defined as sets of conditions that define whether entities are permitted access to a protected resource. The authorization function can support different authorization mechanisms, such as Access Control List (ACL), Role Based Access Control (RBAC), etc. The Authorization function could need to evaluate multiple access control policies in an authorization process in order to get a final access control decision. This process is further described in clause 7.

Authorization evaluation process is based on the Service Subscription resource which specifies what M2M Services and M2M Service roles the authenticated entity has subscribed to and the access control policies associated with the protected resource. The authorization evaluation process can also consider contextual attributes such as time or geographic location.

Prior to authorization mutual authentication between the originator CSE or AE and hosting CSE can be performed as specified in clause 8. Clause 6.1.2.2.1 describes the conditions under which mutual authentication is mandatory. An access control rule can also include an indicator that the access control rule applies only when mutual authentication has been performed successfully and the result of mutual authentication is still current; see clause 7.1.3 for details.

5.1.3 Identity Management

The Identity Management function provides oneM2M identities/identifiers to the requesting entity in case those identities are stored within the secure environment. oneM2M identifiers as defined in the oneM2M Architecture (ETSI TS 118 101 [1]) can also be treated as sensitive data that are accessible to AEs or CSEs and used independently of Authentication or Authorization functions.

5.2 Security Layers

5.2.1 Security Service Layer

The security service layer provides the following services:

- Access Management:
  - Identification and Authentication.

  - Authorization.
  - Access Control.
- Sensitive Data Handling:
  - Sensitive Functions protection.
  - Secure Storage.
- Trust Enabling Security Functions:
  - MEF (M2M Enrolment Function)
  - MAF (M2M Authentication Function)
  - DAS (Dynamic Authorization System)
  - PPM (Privacy Policy Manager)
- Security Association Establishment:
  - Secure Connection via secure session establishment.
  - Secure Connection via object security.
- Security Administration (including remote security provisioning).
- Identity Protection.

Each of these services provides functions and resources on the Security Service and Administration API.

5.2.2 Secure Environment Abstraction Layer

The Secure Environment Abstraction Layer (not specified in the present document) provides access to the Secure Environment via a general Security Transport API. A Plug-in associated to the type of Secure Environment provides physical/logical connectivity to the secure environment. The Secure Environment Abstraction Layer also has to be accessible on the Service Layer.

5.3 Integration within overall oneM2M architecture

Security services are provided within the following architectural components and interact on the different reference points as described in ETSI TS 118 101 [1].

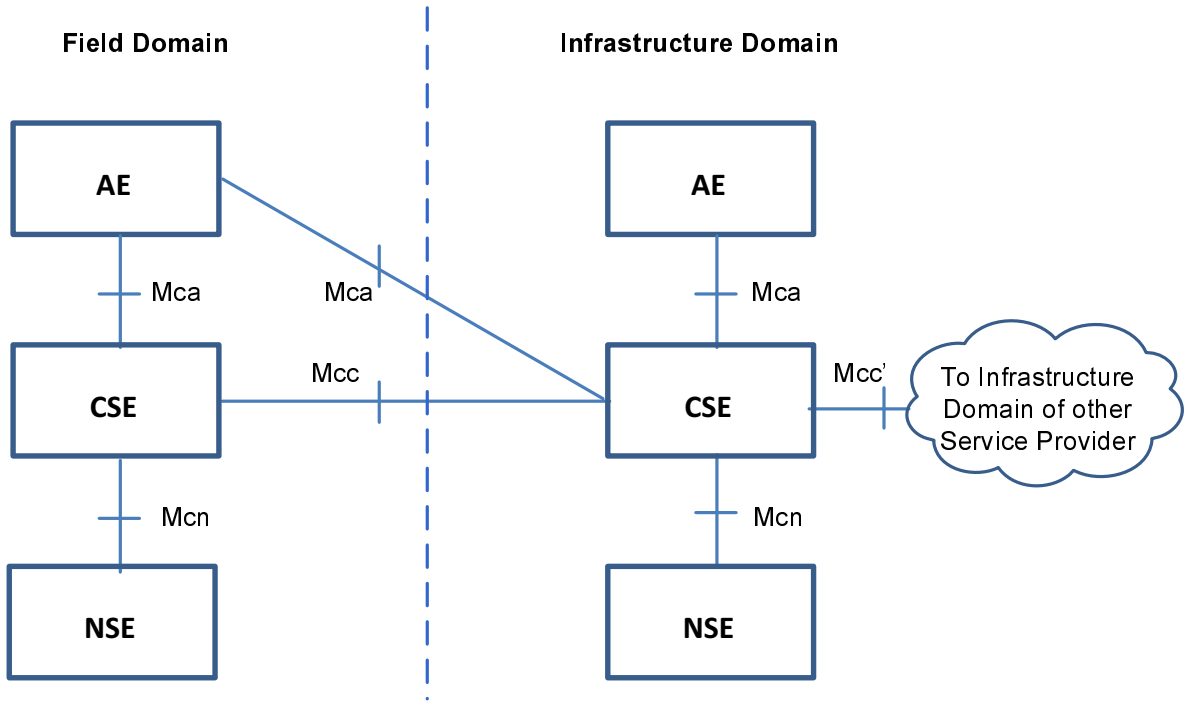


Figure 5.3-1: oneM2M Functional Architecture

6 Security Services and Interactions

6.1 Security Integration in oneM2M flow of events

6.1.1 Interactions between layers

Before any M2M Common Services layer procedure can take place, connectivity has to be established in the underlying Network Services Layer, which may involve independent provisioning and service registration procedures specified by the underlying network.

The Service Layer Security provisioning (security pre-provisioning or security bootstrapping) and Security Association Establishment procedures specified in the present document can take place independently (and generally consecutively) from any required Network Service Layer connectivity establishment procedures.

Finally, the security provisioning and security association establishment requirements imposed by M2M Application Service Providers have to be accounted for. At the service layer level, the security association establishment results in a TLS or DTLS session which protects messages being exchanged between adjacent AE/CSE, i.e. hop-by-hop. AEs that need to preserve the privacy of their information exchange from untrusted intermediate nodes can be provisioned to support a direct security association between them. Such security associations enable to encrypt the content of resources exchanged between AEs through the service layer. In some scenarios (see clause 8.2.1), security association establishment between adjacent AE/CSE requires separate TLS or DTLS sessions for each transmission direction, i.e. a pair of security associations.

6.1.2 High level sequence of events

6.1.2.1 Enrolment phase

M2M equipment typically requires provisioning and configuration phases before being put in actual operation. This can be performed by a pre-provisioning that can be integrated in the manufacturing or product deployment phase, or by means of a security bootstrapping procedure (i.e. remote security provisioning) that takes place before the equipment starts actual operation.

At the service layer level, such provisioning and configuration requires selection of the stakeholder that will provide services through the equipment, especially the M2M Service Provider.

Enrolment phase may occur several times during the lifecycle of an M2M equipment, but is only repeated when a change in the Service Provider affects the provisioning or configuration of the equipment.

The security provisioning phase for the different layers can be combined using a common method of security pre-provisioning.

Remote Security Provisioning Frameworks (RSPF) provides post-provisioning of the essential information to establish a security association between a Field Domain entity and the M2M Authentication Function of a chosen M2M Service Provider. The essential security information includes the security credentials and identifiers. Remote Security Provisioning procedures rely on an M2M Enrolment Function which can be external to the M2M Service Provider to establish appropriate credentials:

- **Pre-Provisioned Symmetric Enrolee Key Remote Security Provisioning Framework:** A symmetric key is pre-provisioned to the Enrolee and M2M Enrolment Function for the mutual authentication of those entities. For more details, see clause 8.3.2.1.
- **Certificate-Based Remote Security Provisioning Framework:** The Enrolee and M2M Enrolment Function are each issued and authenticate themselves with private signing keys and Certificates containing the corresponding Public Verification Key. For more details see clause 8.3.2.2.
- **GBA-based Remote Security Provisioning Framework**. In this case, the M2M Enrolment Function includes the functionality of a GBA Bootstrap Server Function. This framework uses 3GPP or 3GPP2 symmetric keys to authenticate the Enrolee and the M2M Enrolment Function (which is also a GBA BSF). The details are specified by ETSI TS 133 220 [13] and TIA TIA-1098 [14]. For more details see clause 8.3.2.3.

Figure 6.1.2.1-1 illustrates the different Remote Security Provisioning Frameworks. Note there is no communication between M2M Entities A and B in the Remote Security Provisioning procedure. After successful completion of the Remote Security Provisioning procedure, a Security Association Establishment procedure is applied.

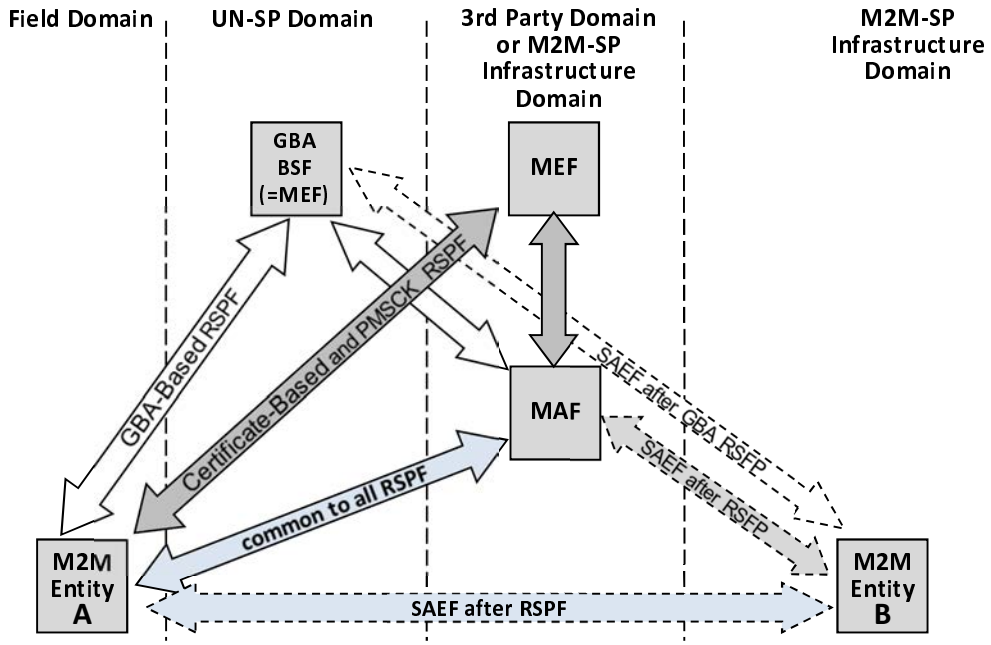


Figure 6.1.2.1-1: Entities involved in Remote Security Provisioning

6.1.2.2 Operational phase

6.1.2.2.1 M2M Service Access

M2M services are offered by CSEs to AEs and/or other CSEs. To be able to use M2M services offered by one CSE, the AEs and/or CSEs need to be mutually identified and authenticated with that CSE, in order to provide protection from unauthorized access and Denial of Service attacks. This mutual authentication enables to additionally provide encryption and integrity protection for the exchange of messages across a single Mca, Mcc or Mcc' reference point. In addition, communicating AEs that require similar protection for their own information exchanges can be provisioned to apply the same security method to their communications.

This is the purpose of the Security Association Establishment procedure, which needs to be executed before the service related procedures specified in ETSI TS 118 101 [1] for the corresponding reference point.

On the Mca and Mcc reference points, security association establishment between a field domain AE or CSE, respectively, and an IN-CSE is mandatory.

On the Mcc' reference point, security association establishment between IN-CSE and IN-CSE is mandatory.

On the Mca reference point, security association establishment between AE and the CSE in the field domain is strongly recommended.

NOTE: Security Association Establishment on the Mca interface in a local domain is optional depending on risk assessment, for instance in scenarios where unauthorized access can be prevented by other security measures out of scope of the present document. This includes the following use cases:

- AE and CSE (i.e. Mca end-points) are securely integrated on the same physical device (i.e. an ASN).
- Secure communication is guaranteed by the Underlying Network (e.g. WLAN or VPN).
- Mca communication takes place on a wire (e.g. Ethernet) in a safe physical environment.

The security association establishment phase of the M2M Service Layer and M2M Application Layer are generally independent from similar procedures possibly required by the Network Layer, though they can rely on the security services provided by the Network Layer.

The oneM2M system supports the following authentication mechanisms for Security Association Establishment, described in more detail in clause 8.2.1:

- **Provisioned Symmetric Key Security Association Establishment Framework:** A symmetric key is pre-provisioned to the Security Association end-points. For more details see clause 8.2.2.1.
- **Certificate-Based Security Association Establishment Framework:** Security Association end-points authenticate themselves using private signing keys and Certificates containing the corresponding Public Verification Key. For more details see clause 8.2.2.2.
- **M2M Authentication Function (MAF) Security Association Establishment Framework:** For MAF-based SAEF, the centralized key distribution server is a MAF hosted either by a 3rd party service provider which has a service relationship with the M2M Service Provider (M2M-SP), or hosted by the M2M-SP itself. The MAF authenticates a Field Domain entity on behalf of an IN-CSE using a symmetric key. For more details see clause 8.2.2.3.

Figure 6.1.2.2.1-1 illustrates the different use cases and entities involved in the various Security Association Establishment Frameworks (SAEF) considered in the present document.

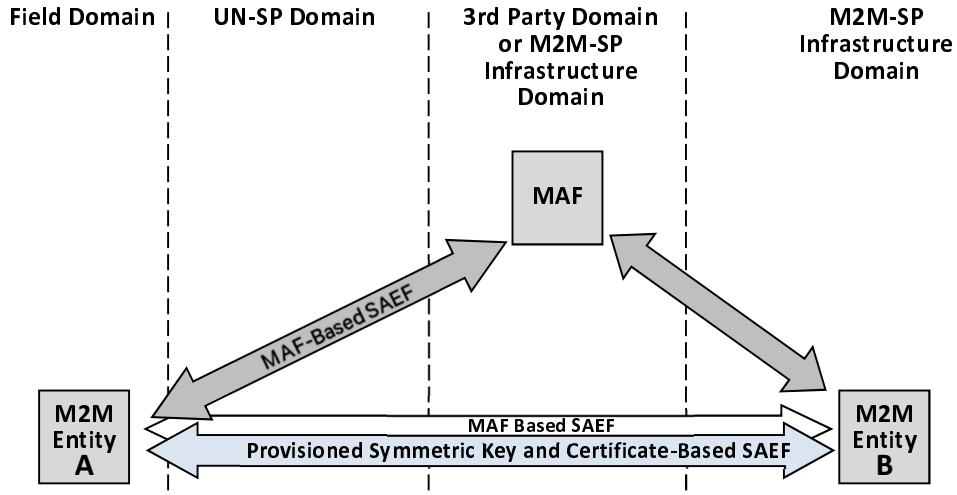


Figure 6.1.2.2.1-1: Entities involved in Security Association Establishment

Once a security association is established between a Registrar CSE and a Registree AE, the resulting secure session may be used to secure messages that are exchanged between one or more M2M Service Users of the Registree AE and the Registrar CSE for the purpose of authenticating the M2M Service Users associated with the Registree AE. Authentication of an M2M Service User allows the Registrar CSE to verify that the M2M Service User knows its M2M-User-ID and corresponding credential.

There are many well-known methods for performing user-based authentication. Some examples, include HTTP username:password based authentication, LDAP based authentication, multifactor authentication, etc. The selection of which method to use typically depends on deployment requirements. For this reason, the definition of this message exchange is out of scope of the present document.

Once a M2M Service User is authenticated and its M2M-User-ID is trusted, the Registrar CSE shall check whether the M2M Service User is authorized to access the Registrar CSE via the Registree AE's established security association. The method to perform this check is specified in ETSI TS 118 101 [1].

6.1.2.2.2 Authorization to access M2M resources

Once an AE or CSE has been granted access to M2M services, the Access Control decision procedure specified in clause 7.1.5 of the present document is executed before accessing an M2M resource, as specified in ETSI TS 118 101 [1].

6.1.2.2.3 Security for multicast group fanout procedures

Multicast group fan out is specified in ETSI TS 118 101 [1]. When this procedure is employed, request primitives originating from a Group Hosting CSE and received by Member Hosting CSEs are transferred on the Mcc reference point over a multicast capable underlying network such as 3GPP MBMS (ETSI TS 123 246 [77]).

The multicast group fan out procedure is applicable only in conjunction with the CoAP binding protocol specified in ETSI TS 118 108 [76]. However, DTLS-based Security Association Establishment cannot be used for multicast. When multicast group fanout over 3GPP MBMS is employed, security as defined in ETSI TS 133 246 [78] shall be applied. Security between Group Hosting CSE and BM-SC is not in the scope of the present document.

The present document does not specify a specific security mechanism on the oneM2M Service Layer (reference point Mcc).

6.2 Security Service Layer

6.2.1 Access Management

6.2.1.1 Identification and Authentication

This component provides authentication services to the Application Layer. Annex B provides a general description of Authentication mechanisms. oneM2M mutual authentication schemes allow oneM2M entities to prove that they know related credentials such as Master Credentials, without having to exchange value of those credentials, and sensitive data such as security identities and security identifiers. To prevent reading and copying of credentials, a secure environment within the Security CSF provides protection against tampering of those credentials and related processed information. For more information see annex B.

6.2.2 Authorization Architecture

Figure 6.2.2-1 provides a high level overview of a generic authorization architecture. This architecture comprises four subcomponents that are described as follows:

- Policy Enforcement Point (PEP):
  - PEP intercepts resource access requests, makes access control decision requests, and enforces access control decisions. The PEP coexists with the entity that needs authorization services.
- Policy Retrieval Point (PRP):
  - PRP obtains applicable authorization policies according to an access control decision request. These applicable policies should be combined in order to get a finial access control decision. The PRP is located in the Authorization service.
- Policy Information Point (PIP):
  - PIP provides attributes that are needed for evaluating authorization policies, for example the IP address of the requester, creation time of the resource, current time or location information of the requester. The PIP is located in the Authorization service.
- Policy Decision Point (PDP):
  - PDP interacts with the PRP and PIP to get applicable authorization polices and attributes needed for evaluating authorization policies respectively, and then evaluates access request using authorization policies for rendering an access control decision. The PDP is located in the Authorization service.

The Authorization service can comprise any of the subcomponents: PDP, PRP and/or PIP. This means that the subcomponents PEP, PRP, PDP and PIP could be distributed across different nodes. For example the PEP is located in an ASN/MN and the PDP is located in the IN.

The present release supports separation of PRP and PIP on different CSE from PDP as detailed in clause 7.5. The generic procedure described below is provided for information and to support further extensions, while clause 7 provides the details of authorization mechanisms in the current release.

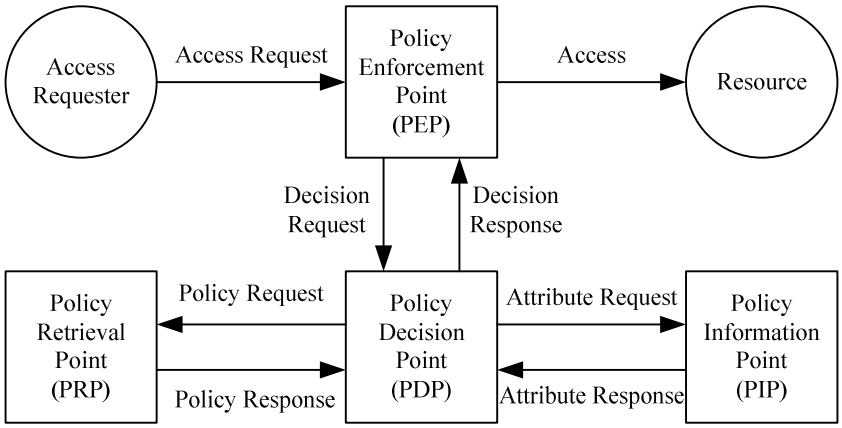


Figure 6.2.2-1: Overview of the authorization architecture

The generic authorization procedure is shown in figure 6.2.2-2.

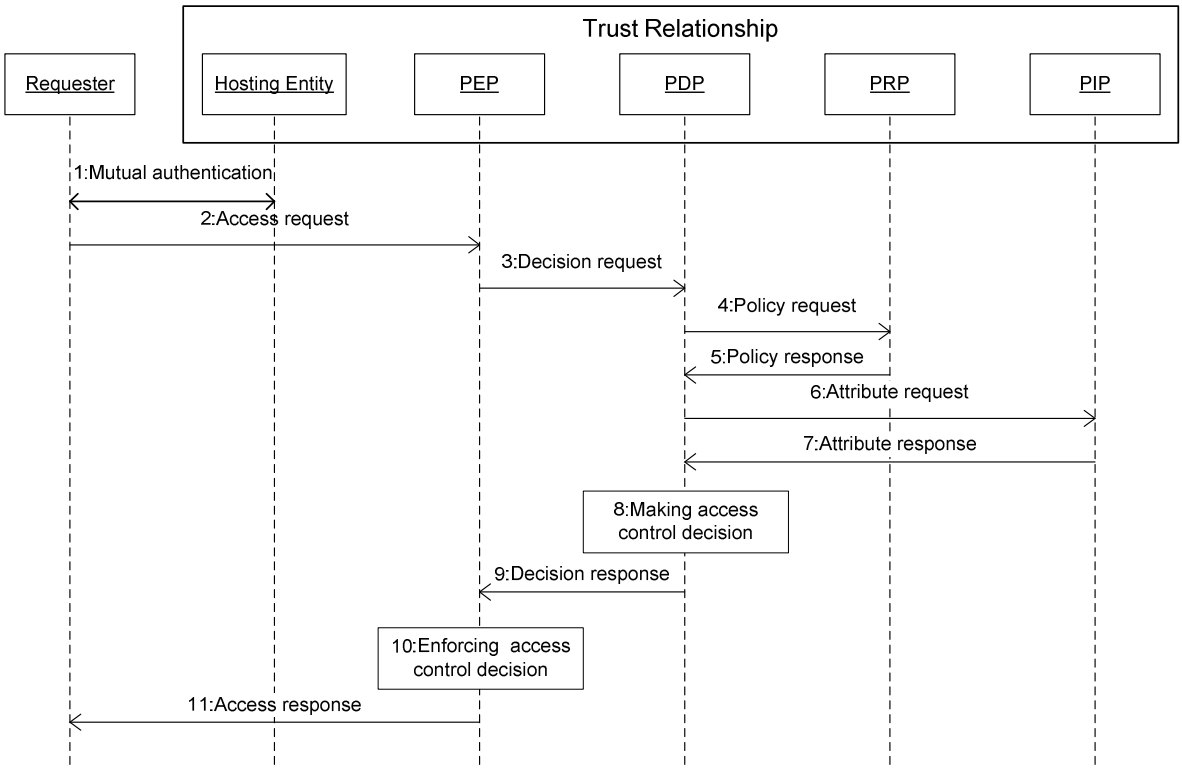


Figure 6.2.2-2: Authorization Procedure

Step 1: Mutual authentication (Pre-requisite).

Step 2: Access Requester sends an Access Request to the PEP.

Step 3: PEP makes an Access Control Decision Request according to the requester's Access Request, and sends the Access Control Decision Request to the PDP.

Step 4: PDP sends an Access Control Policy Request that is generated based on the Access Control Decision Request to the PRP.

Step 5: PRP finds all applicable access control policies to the access request and sends them back to the PDP. When multiple access control policies are involved, the PRP also provides a policy combination algorithm for combining multiple evaluation results into one finial result.

Step 6 PDP sends Attribute Request to the PIP if any attributes are required for evaluating these access control policies.

Step 7: PIP gets required attributes and sends them back to the PDP.

Step 8: PDP evaluates Access Request using access control policies. When there are multiple applicable access control policies, the PEP needs to calculate a final Access Control Decision using the policy combination algorithm.

Step 9: PDP returns the Access Control Decision back to the PEP.

Step 10: PEP enforces the access control decision, i.e. either forwards the Access Request to the resource or denies this access.

Step 11: PEP returns access result back to the Access Requester.

6.2.3 Security Administration

6.2.3.0 Introduction

The Security Administration service provides the capability to manage the Security functions, resources and attributes. This includes management of resources provided via the secure environment. In addition it can provide functions to manage sensitive data with their associated identifiers and subscriptions on behalf of other entities. Security administration is therefore dependent upon the type of secure environment being used (independent hardware module, integrated trusted execution environment or software protection). Depending on the type of Secure Environment, distinct existing standards can be used for remote administration of those Secure Environments.

6.2.3.1 Security Pre-Provisioning of SE

Several sensitive data and associated objects are often configured by pre-provisioning of a secure environment (see clause 6.3.1) prior to deploying the M2M device it is associated with.

UICCs specified in ETSI TS 102 671 [23] and ETSI TS 102 221 [24] are commonly used for such purpose because their use is required to access some underlying networks, they provide a high security level, and they offer an interoperable transport interface specified in ETSI TS 102 221 [24]. UICC-based oneM2M pre-provisioning shall follow the framework specified in annex D to ensure interoperability.

For asymmetric security schemes relying on public / private key pairs, the interoperable framework to interface an M2M device with a secure environment hardware supporting generation of asymmetric key pairs, described in annex L, may be supported, so that private keys are never exposed outside of the secure environment.

6.2.3.2 Remote security administration of SE

Security sensitive data and functions that are protected and isolated within the SE may remain remotely accessible to legitimate security administrators after deployment. Remote security administration differs from standard device management by the expectation that a secure channel is intended to be established between the administration server and the Secure Environment of the M2M Node (i.e. the secret used to secure the connection is not available in the M2M node outside of the Secure Environment). Applicable remote security administration protocols are dependent on the risk level of each M2M application and not just on the underlying network technologies. Widespread technologies that enable remote security administration for the different security levels distinguished in ETSI TR 118 508 [i.4] are considered in annex C.

Since remote security administration requires the target sensitive information to be remotely modifiable, protection of such sensitive information from remote software hacking of the device is particularly critical. In case the Secure Environment relies on software protection only, remote security administration of the following data should be allowed only where remote access by potential attackers can be mitigated:

- Private key and associated identifiers.

- Long-term shared symmetric key (compared to expected lifetime of the M2M node) and associated identifiers.
- Any process and parameters thereof that manipulates the above information, i.e. security functions.

6.2.4 Identity Protection

Identity Protection provides services to the Application Layer such as pseudonyms and protecting the anonymity of transactions.

6.2.5 Sensitive Data Handling

6.2.5.0 Introduction

The Sensitive Data Handling service provides certain Sensitive Functions to the Application Layer.

Sensitive Functions comprise the following functions:

- Secure Storage.
- Cryptographic operations.
- Methods for bootstrapping initial secrets (e.g. GBA symmetric key derivation supported in annex D, or generation of asymmetric key pairs in a secure environment as specified in annex L).

6.2.5.1 Sensitive Functions

This service provides AEs and CSEs with access to Sensitive Functions of the SE.

6.2.5.2 Secure Storage

This service provides AEs and CSEs with access to the secure storage capability of the SE. Data securely stored by the AE or CSE is intended to be accessible only through the Security API and by authorized entities. Secure Storage should be managed by the Secure Environment. Securely stored data is intended to remain under the control of the stakeholder owning the data, i.e. the entity that requested the data to be stored within the secure storage, independently of other stakeholders.

6.2.6 Trust Enabling security functions

oneM2M Trust Enabling Architecture may require the presence of security functionalities within the Infrastructure Domain: an M2M Authentication Function (MAF) and an M2M Enrolment Function (MEF), both classified as Trust Enabling Functions (TEF) and serving authentication and end-to-end security purposes, as well as Dynamic Authorization System (DAS) server or Role Authorities serving authorization purposes. The M2M Authentication Function and the M2M Enrolment Functions shall incorporate the ability to provide for End-to-End credential registration and provisioning. In addition, a Privacy Policy Manager functionality (PPM) may be implemented to protect user's privacy. All of these functions can be either under M2M Service Provider control or delegated to a M2M Trust Enabler (i.e. a party trusted by all involved M2M ecosystem stakeholders).

- M2M Enrolment Function (MEF):
  - The MEF is used during the enrolment phase and supports the security bootstrap procedure enabling the provisioning of the Master Credentials to be used to mutually authenticate entities accessing the infrastructure of an M2M Service Provider. The MEF relies on an initial credential pre-provisioned in the M2M node (e.g. during manufacturing).
  - The credentials provisioned by an MEF can be used for authentication with an M2M Authentication Function in the MAF-Based Security Association Establishment Framework (SAEF), End-to-End Security of Primitives (ESPrim) or End-to-End Security of Data (ESData). Alternatively, the provisioned credentials may be used directly in the SAEF, ESPrim or ESData.

- M2M Authentication Function (MAF), used during the operational phase of M2M Services:
  - Master Credentials, used to mutually authenticate CSEs/AEs during the operation phase, are securely stored in a specific infrastructure functionality named M2M Authentication Function (MAF).
  - The MAF securely contains the set of Master Credentials that are used for authenticating CSEs/AEs that have been enrolled through the M2M SP or M2M Trust Enabler. The MAF stores the Master Credentials and possibly the identifiers of the associated CSE/AE.
  - A single MAF may support all communication security services (SAEF, ESPrim and ESData) or only a selection of them. An MAF providing MAF-based SAEF is operated by the M2M SP, or by an M2M Trust Enabler on behalf of the M2M SP. Other MAF can be operated by M2M Trust Enabler or M2M SP, and there is no assumption of a trust relationship existing between the M2M Trust Enabler and M2M SP in those cases.
  - The MAF is also in charge of all security operations involving the usage of the Master Credentials.
- Dynamic Authorization System (DAS) server and Role Authorities: These functionalities manage authorization privileges to access resources that may be assigned during operation and are described in clauses 7.3 and 7.4, respectively.
- Privacy Policy Manager (PPM): This functionality assists in the management of privacy preferences expressed by data subject with respect to service requirements and applicable regulations, and is described in clause 11.

6.3 Secure Environment Abstraction Layer Components

6.3.1 Secure Environment

The Secure Environment component is an entity that provides Sensitive Functions operating on Sensitive Data, Secure Storage and other resources/functions.

The security sensitive data and security functions contained in M2M field domain nodes are intended to be protected from unauthorized access or alteration, as determined by risk analysis. Sensitive data and functions include security credentials and algorithms that manipulate them. The purpose of a Secure Environment is to provide the required protection level (see table 6.3.1-1) to sensitive data during storage and usage, including primarily any long term symmetric or asymmetric cryptographic secret used during operation. Additionally, isolation of security sensitive data and functions controlled by different stakeholders within an M2M node can be ensured by distinct secure environments. This is especially critical for M2M Nodes that can be remotely or physically accessed by potential attackers.

The choice of a Secure Environment is guided by a risk analysis considering all layers of an M2M application, though it should leverage where possible on capabilities provided by the M2M Service Layer or the Underlying Network, e.g. UICC in 3GPP and 3GPP2 networks, or Trusted Execution Environment requirements.

There is no assumption made on the particular implementation of the Secure Environment. A SE may be implemented as an independent HW Secure Element or as an integrated SW function. Each Secure Environment can be associated with one certain Security Level depending on the particular implementation of the SE. Different Secure Environments provide different Security Levels and protection levels as indicated in table 6.3.1-1.

Table 6.3.1-1: Classification of Protection levels

Protection Level	Description
0	No protection. The data are exposed even without active attacks.
1	Low protection, data are protected from passive observers but could be exposed by active attacks, be they local or remote. E.g. software solutions exist that rely on general purpose processing hardware of the supporting equipment.
2	Medium protection, protection of the data from remote attacks is addressed, but local attacks, especially physical attacks, remain possible, i.e. Medium protection provides countermeasures against software attacks only. E.g. Software solutions to protect data and sensitive functions rely on specific processing providing enforced isolation and enables sensitive code and data to be kept away from an unprotected operating environment, software and memory. The code running in the protected environment is cryptographically verified for integrity assurance.
3	High protection, addressing both remote and local attacks to access the data, including attacks involving physical access. This includes strong counter measures against software and hardware attacks, such as detection of abnormal operating conditions and scrambling plus hardware masking of the memory and side channel analysis of operations involving sensitive data.

There is intended to be at least one Secure Environment in each M2M node providing secure storage to the local CSEs and AEs, however there could be multiple.

6.3.2 SE Plug-in

The SE Plug-in enables physical access to the respective Secure Environment. Depending on the type of Secure Environment, the SE Plug-in can be implemented differently for each Secure Environment.

NOTE: Specification of the SE Plug-in is out of scope of the present document.

6.3.3 Secure Environment Abstraction

This component is specified in ETSI TS 118 116 [67].

7 Authorization

7.1 Access Control Mechanism

7.1.1 General Description

The M2M authorization procedure controls access to resources and services hosted by CSEs and AEs. The authorization procedure requires that the originator of the resource access request message has been identified to the Authentication Function, and originator and receiver are mutually authenticated with each other.

The resource addressed in a request message has an associated *accessControlPolicyIDs* attribute (either included explicitly as an attribute of the resource addressed in the request message, implied from the parent of the resource, or set fixed by the system, see clause 9.6.1 of ETSI TS 118 101 [1]). The *accessControlPolicyIDs* attribute contains a list of identifiers of <*accessControlPolicy*> resources applicable to the resource addressed in the request message.

The overall structure of <*accessControlPolicy*> resources is described in clause 9.6.2 of ETSI TS 118 101 [1]).

Each of these <*accessControlPolicy*> resources include *privileges* and *selfPrivileges* attributes, which comprise the information, denoted as *access control rules* in the present document, that is evaluated against the parameters associated with the request message to obtain the access decision.

Figure 7.1.1-1 illustrates the relation between <*accessControlPolicy*> resource instances (ACP) and the instances of the protected resources, denoted Resource_1 to Resource_N.

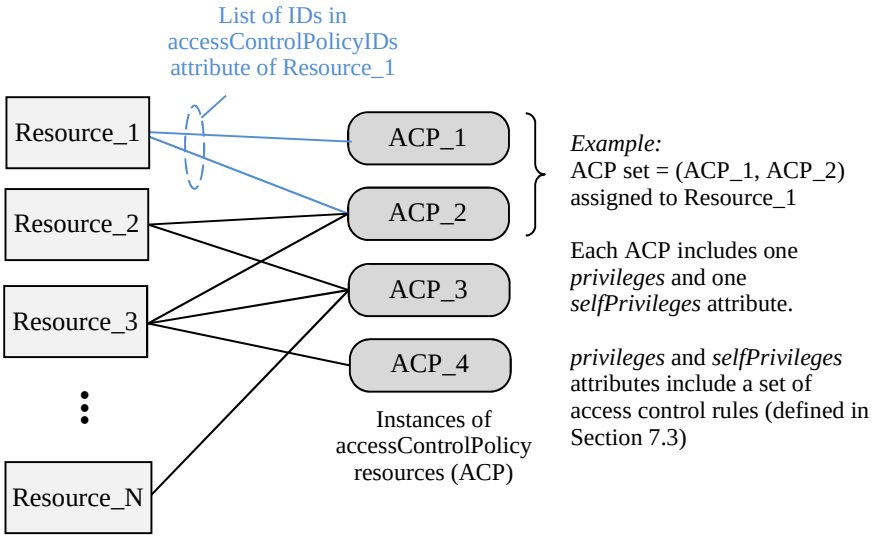


Figure 7.1.1-1: Relation between Resource Instances and Access Control Policies

Access requests to ACP's itself are evaluated against the *selfPrivileges* attribute of that ACP. Access requests to instances of all other resource types, are evaluated against the *privileges* attributes of the ACP set associated with the targeted resource.

For requests to <*accessControlPolicy*> resource type, authorization is granted if the request is evaluated to "Permit" for at least one *selfPrivileges* attribute. For other resource types, authorization is granted if the request is evaluated to "Permit" for at least one *privileges* attribute.

The *privileges* and *selfPrivileges* defined in the *accessControlPolicy* resource determine *which request originator* is allowed to access the resource containing this attribute, for *which specific operation* (i.e. Create, Retrieve, Update, Delete, etc.) and *for which specific context constraints* (i.e. constraints regarding access time, originator's IP address and originator's location).

The access control approach specified here conforms to the concept of Attribute Based Access Control (ABAC) as defined in [i.12].

The policies defined in the <*accessControlPolicy*> resources are enforced by an access control mechanism which employs the authorization logical architecture outlined in clause 6.2.2.

The access control mechanism assembles the information needed to render the access decision which consists of:

- Information included in the resource access request message as defined in clause 7.1.2 (table 7.1.2-1).
- Contextual information as defined in clause 7.1.2 (table 7.1.2-2).
- Tokens (if any) associated with the resource access request.
- The policies governing the access as defined in clause 7.1.3.

7.1.2 Parameters of the Request message

This clause specifies the parameters of a request message which are evaluated by the access control mechanism.

The data types applicable to these parameters are defined in clause 6.4 of ETSI TS 118 104 [4].

The parameters are listed in table 7.1.2-1.

Table 7.1.2-1: Parameters indicated in the request message

Parameter	Description	Mandatory/ Optional	Usage in access control mechanism
To	URI of target resource	M	Selection of accessControlPolicy associated with the target resource
From	Identifier representing the originator of the request	M (see note 1)	Evaluated against accessControlOriginators in *privileges* and *selfPrivileges* attributes
Role IDs	Role IDs of the originator	O	Evaluated against accessControlOriginators in *privileges* and *selfPrivileges* attributes
Operation	Requested operation	M	Evaluated against accessControlOperations in *privileges* and *selfPrivileges* attributes
Resource Type	Type of the target resource	O (see note 2)	Evaluated against accessControlObjectDetails in *privileges* attributes. Applicable to Create operations only.
Filter Criteria	***filterUsage*** condition tag in Filter criteria	O	Differentiation between Retrieve and Discovery operations
Tokens	ESData-protected Tokens	O	Contains authorization information (e.g. Role-IDs) to be used in the decision for the request
Token IDs	tokenIDs **or** Local-Token-ID	O	Identifies Tokens containing authorization information (e.g. Role-IDs) to be used in the decision for the request
M2M Service User	Identity of a M2M Service User	O	Evaluated against the accessControlUserIDs sub-parameter of the accessControlContexts parameter of the *privileges* and *selfPrivileges* attributes
NOTE 1: The ***From*** primitive parameter is Mandatory in all requests except for AE registration procedure where it is optional, as specified in ETSI TS 118 101 [1].			
NOTE 2: The ***resource Type*** primitive parameter is present in Create request primitives only.			

Table 7.1.2-2 lists the context parameters associated with a request message which are evaluated by the access control mechanism. These parameters are not explicitly included in a request message but can be obtained at the receiver and validated against the context policy parameters as given in table 7.1.2-2.

Table 7.1.2-2: Context parameters associated with a request message

Parameter	Description	Usage in access control mechanism
rq_time	Time stamp when the request message was received at the hosting CSE. Obtained by the hosting CSE's system time clock.	Validated against accessControlTimeWindow parameter in an access control rule, see clause 7.1.3.
rq_loc	Location information about the originator of the request. Obtained over the Mcn reference point.	Validated against accessControlLocationRegion parameter in an access control rule, see clause 7.1.3.
rq_ip	IP source address associated with the IP packets that carry the request message. Obtained over the Mcn reference point.	Validated against accessControlIpAddress parameter in an access control rule, see clause 7.1.3.

Tokens, as defined in clause 7.3.2.4, may be associated with a request message. A Token may be associated with a request as a result of being included in the ***Tokens*** primitive parameter of the request message or identified in the ***Token IDs*** primitive parameter of the request message. If the Hosting CSE obtained a token from the Dynamic Authorization System (DAS) Server using Direct Dynamic Authorization, then this Token shall be associated with a request if the holder parameter in the Token matches the Absolute AE-ID or CSE-ID of the Originator of the request. Dynamic Authorization is specified in clause 7.3.

Table 7.1.2-3 lists the security context parameters associated with a request message.

Table 7.1.2-3: Security Context parameters associated with a request message

Parameter	Description	Mandatory/ Optional	Usage in access control mechanism
rq_authn	Boolean value (TRUE/FALSE) indicating if the Originator is considered to have been authenticated by the Hosting CSE, and the ***From*** parameter matched the authenticated identity of the Originator.	M	Validated against accessControlAuthenticationFlag parameter in an access control rule, see clause 7.1.3.

The following criteria shall be applied to determine if an Originator is considered to have been authenticated by the Hosting CSE.

- If the Originator is an AE registered to the Hosting CSE, then the criteria for deciding whether the Originator is considered authenticated is deployment and/or implementation specific and depends on the trust guaranteed by the field device's physical and logical embodiment bearing the AE(s) and Hosting CSE (e.g. secure boot and tamper resistance). In many cases it is appropriate to expect a secure channel implying authentication (e.g. a TLS or DTLS session) to be used to protect primitives on the Mca interface, in which case the authentication shall be considered valid for the duration of the TLS session, When this is not the case, e.g. because the physical and logical design is trusted, authentication may be considered to be permanently valid unless it is detected that the device is compromised.
- If the Originator is a CSE registered with the Hosting CSE, then the Originator shall be considered authenticated for the duration of a (D)TLS session because the Mcc is always required to be protected by TLS or DTLS according to a Security Association Establishment Framework (SAEF) as described in clause 8.2. The other CSE may be the Registrar or Registree with respect to the Hosting CSE.
- If the Originator is an AE or CSE registered with a CSE other than the Hosting CSE, then the Originator is considered authenticated by the Hosting CSE if and only if the request primitive is protected using End-to-End Security of Primitives (ESPrim) as described in clause 8.4.

7.1.3 Format of *privileges* and *selfPrivileges* Attributes

The *privileges* and *selfPrivileges* attributes exhibit the same data type format which is specified as follows.

Each *privileges* or *selfPrivileges* attribute comprises a set of access control rules. In the following, the set of access control rules is denoted as *acrs* and an individual access control rule in this set as *acr*. The access control rules in *acrs* are indexed with the letter *k*. The number of access control rules in the set is denoted with the letter K:

$$acrs = \{ acr(1), acr(2), ..., acr(k), ..., acr(K) \}$$

Each access control rule *acr*(*k*) is comprised of mandatory accessControlOriginators and accessControlOperations components and optional accessControlContexts, accessControlObjectDetails, accessControlAuthenticationFlag and accessControlAttributes components.

Hence, an access control rule *acr*(*k*) is either represented as a pair:

$$acr(k) = \{acr(k)\_accessControlOriginators, acr(k)\_accessControlOperations\}$$

or as a 3-tuple, 4-tuple, 5-tuple or 6-tuple. For example, a 3-tuple such as the following:

$$acr(k) = \{acr(k)\_accessControlOriginators, acr(k)\_accessControlOperations, acr(k)\_accessControlContexts\}$$

The generic term "access-control-rule-tuple" is used when referring to a rule *acr*(*k*).

A set *acrs* of access control rules may consist of a mix of pairs,3-tuples, 4-tuples, 5-tuples or 6-tuples. For pairs or for any tuples not containing accessControlContexts, any context parameters associated with a request message are admissible.

The six component parameters of an access-control-rule-tuple supported in the present document are shown in table 7.1.3-1.

Table 7.1.3-1: Parameters of an access-control-rule-tuple

Parameter	Usage Description	Mandatory/Optional	Format
accessControlOriginators	Set of Originators that can be authorized	M	List of CSE-IDs and/or AE-IDs, or keyword "all" to grant access to all originators
accessControlOperations	Set of Operations that can be authorized	M	Enumerated list of operations Create Retrieve, Update, Delete, Discover, Notify
accessControlContexts	See table 7.1.3-3	O	See table 7.1.3-3
accessControlObjectDetails	See table 7.1.3-4	O	See table 7.1.3-4
accessControlAuthenticationFlag	Indicates whether the rule applies only to Originators which are considered to be authenticated by the Hosting CSE	O	Boolean
accessControlAttributes	Set of resource attributes for which access can be authorized	O	List of resource attribute name(s).

The accessControlOriginators parameter comprises a list of SP domain names, CSE-IDs, AE-IDs, resource-IDs of <group> resources and/or Role IDs of any format defined in ETSI TS 118 101 [1]. If access for all originators is to be allowed, the reserved keyword "all" may be included into the value space of accessControlOriginators.

Using a SP domain name in accessControlOriginators means all AE-IDs and CSE-IDs matching the given domain name can be authorized.

It is furthermore allowed to use wildcard character "*", in representations of M2M-SP-ID (i.e. SP domain names), CSE-ID and AE-ID. The scope of a "*" is terminated by a following "/" character. Table 7.1.3-2 shows examples of using wildcard characters in CSE-IDs and AE-IDs.

Wildcard characters are not permitted in resource-IDs of <group> resources and Role IDs.

Table 7.1.3-2: Examples of using wildcard characters in CSE-IDs and AE-IDs of accessControlOriginators

	Form of ID	Examples	Meaning
CSE-ID	Absolute	//m2msp.org/myCSEID //*/myCSEID //*/myCSE*	Any CSE whose ID matches the wild cards
	SP-relative	/myCSEID /myCSE*	Any matching CSE from the SP that is hosting the target resource
AE-ID	Absolute	//m2msp.org/S988 //*/myCSEID/C9886 //*/myCSE*/C9886	Any AE whose ID matches the wild cards
	SP-relative	/myCSEID/C9886 /myCSEID/C98* /myCSE*/C98* /SmyAE*	Any matching AE from the SP that is hosting the target resource

The data type applicable to accessControlOriginators is defined in ETSI TS 118 104 [4].

The accessControlOperations parameter comprises a list of admissible operations which can be any subset of the following elements: Create, Retrieve, Update, Delete, Discover, and Notify. While Create, Retrieve, Update, Delete, and Notify operation are explicitly indicated in the *op* parameter of a request message, the Discovery operation is indicated by the *filterUsage* condition of the ***Filter Criteria*** request parameter having a value of "Discovery", "Discovery-based Operation" or "IPE On-Demand Discovery".

The data type applicable to accessControlOperations is defined in ETSI TS 118 104 [4].

The accessControlContexts parameters are listed in table 7.1.3-3.

Table 7.1.3-3: Parameters of accessControlContexts

Parameter	Usage Description	Mandatory/Optional	Formats
accessControlTimeWindow	Set of Time Windows that can be authorized	O	List of time intervals where access can be granted in extended crontab format
accessControlLocationRegion	Set of Location Regions that can be authorized	O	1) Latitude/longitude coordinates, and a radius defining a circular region around the coordinates 2) Country code
accessControlIpAddress	Set of IPv4 and IPv6 addresses that can be authorized	O	IPv4: dotted-decimal notation with CIDR suffix IPv6: colon separated groups of hexadecimal digits with CIDR suffix
accessControlUserIDs	Set of M2M Service Users that can be authorized	O	List of M2M-User-IDs
accessControlEvalCriteria	Set of conditions that are factored into authorization decisions	O	A tuple consisting of a mandatory resource identifier of a subject resource and an set of evaluation criteria applicable to the subject resource.
accessControlLimit	Number of times access to a resource can be authorized	O	A number that indicates how many times access can be granted.

The accessControlTimeWindow parameter represents a list of elements that comply with the extended crontab syntax as defined in clause 7.3.8 of ETSI TS 118 104 [4]. It allows definition of periodically recurring time intervals at which access can be granted, when the ***rq_time*** parameter associated with the access request message falls into such interval.

For the elements of accessControlLocationRegion there are two representation choices. These can be represented by a 2-character country code or a circle with radius *R* centred at a point defined in terms of longitude and latitude parameters. Refer to annex F for detailed information. Each element of accessControlLocationRegion defines an admissible location region, which is compared with the ***rq_loc*** parameter associated with the access request message.

The data types applicable to accessControlLocationRegion and ***rq_loc*** are defined in ETSI TS 118 104 [4].

The accessControlIpAddress parameter represents a list of IPv4 and IPv6 addresses in dotted-decimal notation with CIDR suffix or colon separated groups of hexadecimal digits with CIDR suffix, respectively. If the ***rq_loc*** parameter associated with the access request message matches one of these addresses, access may be granted with regard to this criterion.

The data types applicable to accessControlIpAddress and ***rq_ip*** are defined in ETSI TS 118 104 [4].

The accessControlUserIDs parameter comprises a list of M2M-User-IDs having a format defined in ETSI TS 118 101 [1]. Using just a SP domain name in accessControlUserIDs means all M2M-User-IDs matching the given domain name can be authorized. For example, "//m2msp.org". It is furthermore allowed to use a wildcard character "*" within the SP-Relative-M2M-User-ID portion of a M2M-User-ID. For example, //m2msp.org/homeowner*. A wildcard character is not permitted within the SP domain name portion of a M2M-User-ID.

The data type applicable to accessControlUserIDs is defined in ETSI TS 118 104 [4].

This accessControlEvalCriteria parameter represents the conditions determining if the request operation is to be allowed. It allows conditional access to the resource based on conditions not contained in the received request. The accessControlEvalCriteria parameter is a tuple that consists of a mandatory subjectResourceID element as defined in table 9.6.61-2 in ETSI TS 118 101 [1] and an evalCriteria element defined in table 9.6.61-3 in ETSI TS 118 101 [1].

The accessControlLimit parameter represents the number of times that the policy defined by an access-control-rule-tuple can allow authorization to the requested resource. This attribute maintains the number of remaining accesses allowed. The parameter is decremented each time an access to the requested resource is granted. If the value is greater than zero then access is granted, otherwise access is denied. If the accessControlLimit parameter is not present in an access-control-rule-tuple, then there are no restrictions on the number of times access is granted.

The accessControlAuthenticationFlag parameter is a Boolean value. If the accessControlAuthenticationFlag parameter is not present, then the value is assumed to be FALSE. If the accessControlAuthenticationFlag parameter is TRUE, then this indicates that the access control rule applies only to Originators considered to have been authenticated by the Hosting CSE. Clause 7.1.2 specifies the criteria used to decide whether or not the Originator is considered to have been authenticated by the Hosting CSE.

The accessControlObjectDetails parameters are listed in table 7.1.3-4.

Table 7.1.3-4: Parameters of accessControlObjectDetails

Parameter	Usage Description	Mandatory/Optional	Formats
resourceType	Resource type on which access control rule applies	O	Resource type identifier
specializationID	Identifier of mgmtDefinition or containerDefinition	O	mgmtDefinition or containerDefinition represented as a string.
childResourceType	Set of resource type identifiers that can be created under the parent resource.	O	Resource type list.

The accessControlObjectDetails attribute specifies a subset of child resource types of the targeted resource to which the access control rule applies. If an access control rule includes *accessControlObjectDetails*, then *childResourceType* is specified. An access control rule which does not include any *accessControlObjectDetails* parameters applies to all child resource types of the target resource. The *accessControlObjectDetails* parameter is described in table 9.6.2.4-1 of ETSI TS 118 101 [1]. Child resource types listed in the *childResourceType* component are subject of access control for the Create operation only. Once a child resource is created, the Access Control Policies assigned directly to it apply. The *resourceType* and *specializationID* elements are optional. If either the *resourceType* or *specializationID* element is present in *accessControlObjectDetails*, the CSE matches the type of resource or specialization of the targeted resource with the value specified in the *resourceType* or *specializationID* element. Further checking of *childResourceType* is done only if the *resourceType* or *specializationID* match occurs. However, if the *resourceType* and *specializationID* elements are not provided, then only *childResourceType* match is performed.

The accessControlAttributes attribute specifies a list of one or more resource attribute names. If there is a rule for which all conditions of the rule are satisfied, then other rules shall be ignored. Otherwise, rules that contain *accessControlAttributes* and that satisfy all conditions apart from *accessControlAttributes* are considered to be applicable rules. In this case, the resource attributes associated with the request and its response are evaluated against the union of resource attributes defined across all the *accessControlAttributes* of these applicable rules to determine if access is allowed.

7.1.4 Access Control Decision

The access decision is derived by comparing the parameters associated with a resource access request message as described in clause 7.1.2 with the access control rules included in the *privileges* or *selfPrivileges* attributes of all ACP sets assigned to the protected resource by means of the *accessControlPolicyIDs*, see figure 7.1.1-1.

The result of the access decision algorithm, i.e. the access decision, is the overall result of evaluating the applicable set of access control rules, *acrs*, against the parameters associated with the access request message. This access decision can be represented by a value of binary data type. The overall result of the access decision algorithm is denoted here with the variable name *res_acrs*:

$$res\_acrs = \begin{cases} \text{TRUE or 1} & \text{if the request matches the access control rules} \\ \text{FALSE or 0} & \text{else} \end{cases}$$

The reference access decision algorithm is specified in clause 7.1.5. For any given sets of inputs, an implementation of the access decision processing shall return the same result as the reference access decision algorithm would return for those inputs.

If the access decision algorithm yields the result *res_acrs* = TRUE, then the access decision for the requested resource shall be "Permit".

If the access decision algorithm yields the result *res_acrs* = FALSE, or the access decision algorithm is not capable of deriving a final result (e.g. due to indeterminate parameters), then the access decision for the requested resource shall be "Deny".

The access decision algorithm consists of two phases. The first phase evaluates each of the applicable access control rules on an individual basis to determine whether an individual access control rule permits access to the requested resource. If the access control decision for an individual access control rule results in *res_acrs* = TRUE, the algorithm stops, and the second phase of the algorithm is not performed. However, if the first phase of the reference access decision algorithm results in *res_acrs* = FALSE, but during the processing of the first phase of the algorithm, one or more access-control-rule-tuple including an *accessControlAttributes* condition is processed, then a second phase of the algorithm is performed. In this second phase, the set of access control rules that have *accessControlAttributes* conditions and that have satisfied all conditions of the first phase of the access decision algorithm, apart from their *accessControlAttributes* conditions, are collectively evaluated. If the union of resource attributes defined across all their *accessControlAttributes* conditions permit access to the requested resource then *res_acrs* = TRUE, otherwise *res_acrs* = FALSE.

Note, although the access decision algorithm is defined as having two phases, it is left to implementation whether these two phases require one or more passes over the access control rules.

7.1.5 Description of the Access Decision Algorithm

The reference access decision algorithm specified in this clause combines partial access control results obtained for each of the individual access control rules contained in a *privileges* or *selfPrivileges* attribute. Further, if multiple ACP instances are assigned to the protected resource, the reference access decision algorithm combines the partial access control results obtained for the individual ACPs of an ACP set.

The algorithm specified in this clause adopts a "Permit-overrides" combining algorithm with respect to access control rules and ACPs as defined in XACML [i.5]. This algorithm has the following behaviour:

1) The first phase of the algorithm determines if the result is "Permit" if a single access control rule included in the *privileges* (or *selfPrivileges*) attribute of a single ACP permits access to the targeted resource.

2) The second phase of the algorithm determines if the result is "Permit" if the set of applicable access control rules containing *accessControlAttributes* collectively as a union permit access to the targeted resource.

3) Otherwise, the result is "Deny".

The logic for evaluating a request against a privilege can be described mathematically as follows. A *privileges* or *selfPrivileges* attribute included in an <*accessControlPolicy*> resource represents a set of access control rules, *acrs*, which is built as in figure 7.1.5-1.

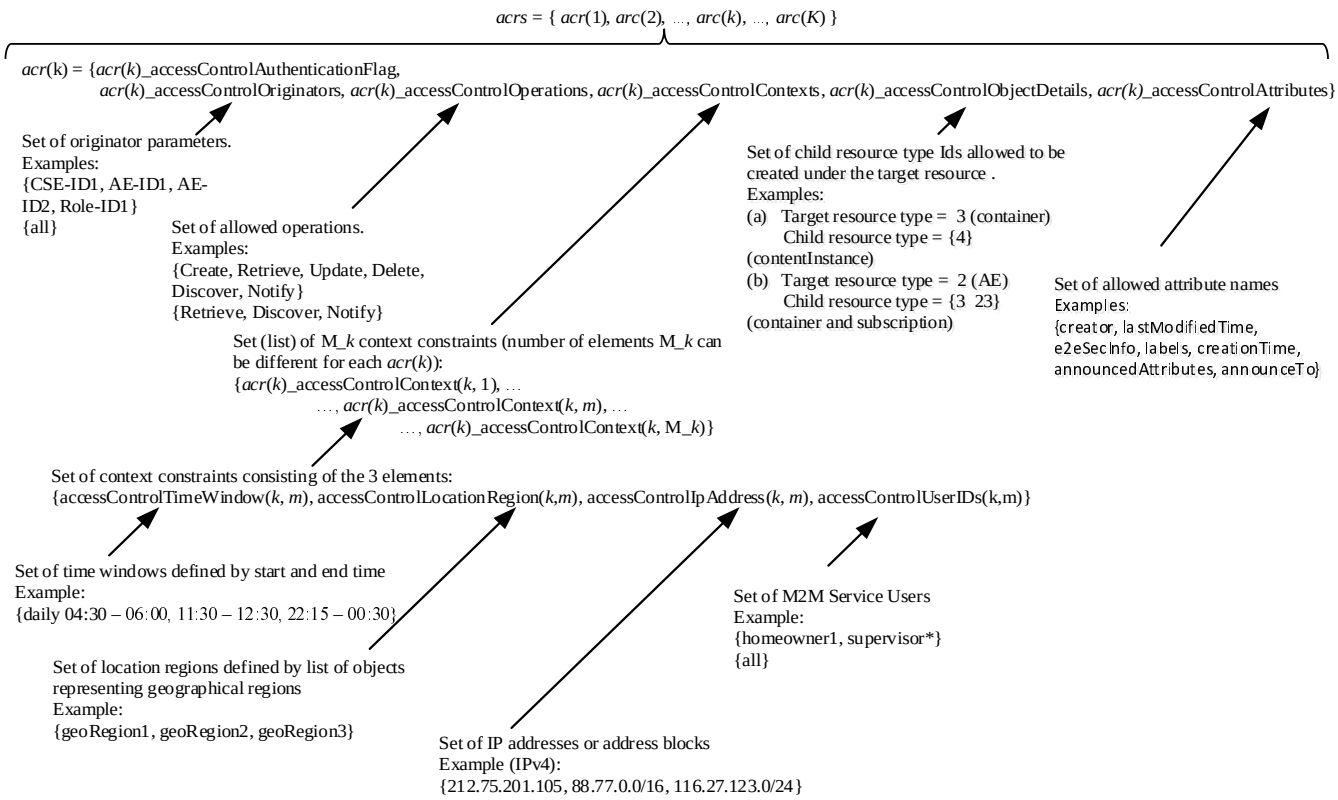


Figure 7.1.5-1: Logic to evaluate privileges in the reference access decision algorithm

The parameters associated with a request, which are evaluated against the parameters contained in the access control rules are specified in clause 7.1.3.

The access decision *res_acrs* defined in clause 7.1.4 is derived by evaluating whether or not the parameters associated with the request message listed in tables 7.1.2-1 and 7.1.2-2 match any of the access control rules contained in the access control rule set defined in clause 7.1.3 as follows:

$$res\_acrs = res\_acr(1) \text{ OR } res\_acr(2) \text{ ... OR } res\_acr(k) \text{ … OR } res\_acr(K),$$

where $res\_acr(k)$ represents the logical evaluation result (i.e. TRUE/FALSE or 1/0) of the request parameters against the k^{th} access control rule in the set *acrs*, which can be expressed as follows:

$$res\_acr(k) = res\_authn(k) \text{ AND } res\_origs(k) \text{ AND } res\_ops(k) \text{ AND } res\_ctxts(k) \text{ AND } res\_objd(k) \text{ AND } res\_attrs(k),$$

where $k = 1 \ldots K$.

The first partial logical result variable *res_authn(k)* on the right side of above equation shall be evaluated according to table 7.1.5-1:

Table 7.1.5-1: Evaluating *res_authn(k)*

***acr(k)*_accessControlAuthenticationFlag**	***rq_authn***	***res_authn***
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	TRUE
FALSE	FALSE	TRUE

The next 4 partial logical result variables on the right side of above equation can be defined by using the following set function:

$$\text{ismember}(\ x, \text{setX}) = \begin{cases} \text{TRUE or 1} & \text{if } x \in \text{setX} \\ \text{FALSE or 0} & \text{else} \end{cases}$$

With this definition:

res_origs(*k*) = ismember(***Originator***, *acr*(*k*)_accessControlOriginators)

res_ops(*k*) = ismember(***Operation***, acr(*k*)_accessControlOperations)

In the above equation, the ***Originator*** variable refers to the authenticated identity of the originator of the request primitive which matches the ***From*** parameter.

The fourth partial logical result *res_ctxts*(*k*) is derived as follows:

res_ctxts(*k*) = *res_context*(*k*, 1) ... OR *res_context*(*k*, *m*) ... OR *res_context*(*k*, M_*k*),

where:

res_context(*k*, *m*) = *res_time*(*k*, *m*) AND *res_ip*(*k*, *m*) AND *res_loc* (*k*, *m*) AND *res_uids(k, m)*, k = 1…K, *m* = 1…M_*k*

and

res_time(*k*, *m*) = ismember(***rq_time***, *acr*(*k*)_accessControlTimeWindow(*m*))

res_ip(*k*, *m*) = ismember(***rq_ip***, *acr*(*k*)_accessControlIpAddress(*m*))

res_loc (*k*, *m*) = ismember(***rq_loc***, *acr*(*k*)_accessControlLocationRegion(*m*))

res_uids(*k*, *m*) = ismember(***M2M Service User***, *acr*(*k*)_accessControlUserIDs(*m*))

The fifth partial logical result *res_objd*(*k*) applies to Create request primitives only and is derived as

res_objd(*k*) = *res_objdetails*(*k*, 1) ... OR *res_objdetails*(*k*, *m*) ... OR *res_objdetails*(*k*, M_*k*),

where:

res_objdetails(*k*, *m*) = *res_resourceType*(*k*, *m*) AND *res_specializationID*(*k*, *m*) AND *res_childResource*(*k*,*m*),

for *m* = 1…M_*k*. The three logical arguments are defined below.

For each given element *acr*(*k*)_accessControlObjectDetails(*m*) in an access control rule determine if the optional *resourceType* parameter is present

resourceType = *acr*(*k*)_accessControlObjectDetails(*m*)/resourceType

Depending on the presence of *resourceType*, *res_resourceType*(*k*, *m*) is derived as

$$\textit{res\_resourceType}(k,m) = \begin{cases} \text{TRUE or 1}, & \text{if not present in } \textit{acr}(k)\text{\_accessControlObjectDetails}(m) \\ \text{TRUE or 1}, & \text{if present and } \textit{resourceType} = \textit{targetResourceTypeID} \\ \text{FALSE or 0}, & \text{if present and } \textit{resourceType} \neq \textit{targetResourceTypeID} \end{cases}$$

where *targetResourceTypeID* is the resource type identifier associated with the resource addressed in the ***To*** parameter of the Create request primitive.

If the value of the *resourceType* element is 13 (<mgmtObject> specialization) or 28 (<flexContainer> specialization>), the optional specializationID element shall also be included in accessControlObjectDetails:

specializationID = *acr*(*k*)_accessControlObjectDetails(*m*)/specializationID

If *specializationID* is present, it shall be matched against the *mgmtDefinition* or *containerDefinition* attributes given in the ***Content*** parameter of the Create request primitive.

$$res\_specializationID(k.m) = \begin{cases} \text{TRUE or 1,} & \text{if } specializationID \text{ not present in } acr(k)\_accessControlObjectDetails(m) \\ \text{TRUE or 1,} & specializationID = mgmtDefinition\ (resourceType = 13) \\ \text{TRUE or 1,} & specializationID = containerDefinition\ (resourceType = 28) \\ \text{FALSE or 0,} & specializationID \neq mgmtDefinition\ (resourceType = 13) \\ \text{FALSE or 0,} & specializationID \neq containerDefinition\ (resourceType = 28) \end{cases}$$

The *childResourceType* element is mandatory in any given accessControlObjectDetails element of an access control rule. It includes a list of $j = 1 \ldots J$ child resource type identifiers to which the rule applies. The j^{th} list element is denoted as follows

childResourceType(*k, m. j*) = *acr*(*k*)_accessControlObjectDetails(*m*)/childResourceType(*j*), *j* = 1…J

The logical variable *res_childResource*(*k, m*) is derived as

res_childResource (*k, m*) = ismember(***Resource Type***, *childResourceType*(*k, m, j*))

where ***Resource Type*** refers to the value of the parameter of the given Create request primitive.

NOTE: If resourceType and specializationID are not present in acr(k)_accessControlObjectDetails(m), res_objdetails(k, m) = res_resourceType(k, m) AND res_specializationID(k, m) AND res_childResource(k,m) = res_childResource(k,m).

The sixth partial logical result *res_attrs*(*k*) is derived as follows:

res_attrs(k)
- TRUE or 1, if the operation specific *acr*(*k*)_accessControlAttributes rules described below are satisfied
- FALSE or 0, if the operation specific *acr*(*k*)_accessControlAttributes rules described below are not satisfied

Depending on the type of operation, the requested attribute names defined within the parameters of the request (e.g. ***To, Content, Filter Criteria***) or within the targeted resource shall be compared against the names of attributes present in *acr(k)*_accessControlAttributes to determine the value of *res_attrs(k)* as follows:

- For an operation that includes a ***Filter Criteria*** parameter and that requires access to the attributes of a resource to process the ***Filter Criteria*** (i.e., matching conditions defined within a discovery operation, discovery-based operation, IPE On-demand discovery operation or a conditional operation), *acr(k)*_accessControlAttributes defines the attributes that can be accessed. If the ***Filter Criteria*** includes the names of attributes that are not defined in *acr(k)*_accessControlAttributes, then then *res_attrs*(*k*) is False or 0. Otherwise, if the names of all the attributes are defined in *acr(k)*_accessControlAttributes, then the value of *res_attrs*(*k*) shall be determined by the operation specific steps described below:
  - For a Retrieve operation, *acr(k)*_accessControlAttributes defines the attributes that can be retrieved and included in the response.
  - For a Retrieve operation that targets a resource, in which all the names of the attributes present in the targeted resource are included in *acr(k)*_accessControlAttributes, then *res_attrs*(*k*) is True or 1. Otherwise, if one or more of the names of the attributes present in the targeted resource are not included in *acr(k)*_accessControlAttributes, then *res_attrs*(*k*) is False or 0.
  - For a Retrieve operation that targets one or more individual attributes of a resource (i.e. partial retrieve) and these attributes are all defined in *acr(k)*_accessControlAttributes, then *res_attrs*(*k*) is True or 1.

Otherwise, if one or more individual attributes are not defined in *acr(k)*_accessControlAttributes, then *res_attrs*(*k*) is False or 0.

- For a Delete operation, *acr(k)*_accessControlAttributes defines the attributes that can be deleted. If all the attributes present in the targeted resource of a Delete operation are defined in *acr(k)*_accessControlAttributes, then *res_attrs*(*k*) is True or 1. Otherwise, if one or more of the attributes are not defined in *acr(k)*_accessControlAttributes, then *res_attrs*(*k*) is False or 0.

- For an Update operation, *acr(k)*_accessControlAttributes defines the attributes that can be included in the ***Content*** parameter of a request and its response. For an Update operation that attempts to create, update or delete one or more attributes of a resource that are all defined in *acr(k)*_accessControlAttributes, then *res_attrs*(*k*) is True or 1, however any attributes of the targeted resource not included in *acr(k)*_accessControlAttributes shall be filtered and not included in the response. Otherwise, if one or more of the attributes of the Update operation are not defined in *acr(k)*_accessControlAttributes, then *res_attrs*(*k*) is False or 0.

- For a Create operation, *acr(k)*_accessControlAttributes defines the attributes that can be included in the ***Content*** parameter of a request and its response. For a Create operation that attempts to create a resource with attributes that are all defined in *acr(k)*_accessControlAttributes, then *res_attrs*(*k*) is True or 1, however any attributes of the targeted resource not included in *acr(k)*_accessControlAttributes shall be filtered and not included in the response. Otherwise, if one or more attributes of the Create operation are not defined in *acr(k)*_accessControlAttributes, then *res_attrs*(*k*) is False or 0.

Thanks to the "Permit-overrides" combining approach, if the access control decision for one access control rule results in *res_acr* = TRUE, the reference access decision algorithm can stop without evaluating any other applicable access control rules of the current ACP or any other ACPs in the ACP set, and the final access decision is "Permit" (i.e. *res_acrs* = TRUE).

However, if the first phase of the reference access decision algorithm results in *res_acrs* = FALSE, and during the processing of the algorithm, one or more access-control-rule-tuple including an *accessControlAttributes* condition is processed, then a second phase of the access decision algorithm shall determine the final access decision. In the second phase of the access decision algorithm, the following steps shall be performed:

- All access control rules having *accessControlAttributes* conditions, that have satisfied all conditions of the access decision algorithm apart from their *accessControlAttributes* condition, shall be collectively considered an applicable set of access control rules,

- Depending on the type of operation, the requested attribute names defined within the parameters of the request (e.g. ***To, Content, Filter Criteria***) or within the targeted resource shall be compared against the names of attributes present in the union of resource attributes defined across all the *accessControlAttributes* of the applicable set of access control rules to determine the value of *res_acrs* as follows:

  - If a Retrieve, Delete, Update or Create operation includes a ***Filter Criteria*** parameter with names of one or more attributes that are not defined in the union of *accessControlAttributes*, then the final access decision shall be "Deny". Otherwise, if the names of the attributes are all defined in the union of *accessControlAttributes*, then the final access decision shall be determined by the operation specific steps described below:

  - For a Retrieve operation that targets a resource, the final access decision shall be "Permit", but any attributes not included in the union of *accessControlAttributes* shall be filtered and not included in the response. If none of the attributes defined in the union of *accessControlAttributes* match the names of the attributes present in the targeted resource, then no attributes shall be returned in the response.

  - For a Retrieve operation that targets one or more individual attributes of a resource (i.e. partial retrieve) and these attributes are all defined in the union of *accessControlAttributes*, then the final access decision shall be "Permit". Otherwise, if one or more individual attributes are not defined in the union of *accessControlAttributes*, then the final access decision shall be "Deny".

  - For a Delete operation, if all the attributes present in the targeted resource are defined in the union of *accessControlAttributes*, then the final access decision shall be "Permit". Otherwise, if one or more of the attributes present in the targeted resource are not defined in union of *accessControlAttributes*, then the final access decision shall be "Deny".

- For an Update operation that attempts to create, update or delete one or more attributes of a resource that are all defined in the union of *accessControlAttributes*, then the final access decision shall be "Permit", however any attributes of the targeted resource not included in the union of *accessControlAttributes* shall be filtered and not included in the response. Otherwise, if one or more of the attributes of the attempted Update operation are not defined in the union of *accessControlAttributes*, then the final access decision shall be "Deny".

- For a Create operation that attempts to create a resource with attributes that are all defined in the union of *accessControlAttributes*, then the final access decision shall be "Permit", however any attributes of the targeted resource not included in the union of *accessControlAttributes* shall be filtered and not included in the response. Otherwise, if one or more attributes of the attempted Create operation are not defined in the union of *accessControlAttributes*, then the final access decision shall be "Deny".

7.2 Impersonation Prevention

7.2.1 Registrar verification of AE-ID

Since several AEs can behave maliciously and pretend to be another AE with their ID changed, the Hosting CSE needs prevention mechanism for AE impersonation. This mechanism works at Registrar CSE since Registrar CSE is an entry point of M2M system.

When the Registrar CSE receives a request, the Registrar CSE shall perform the following procedure.

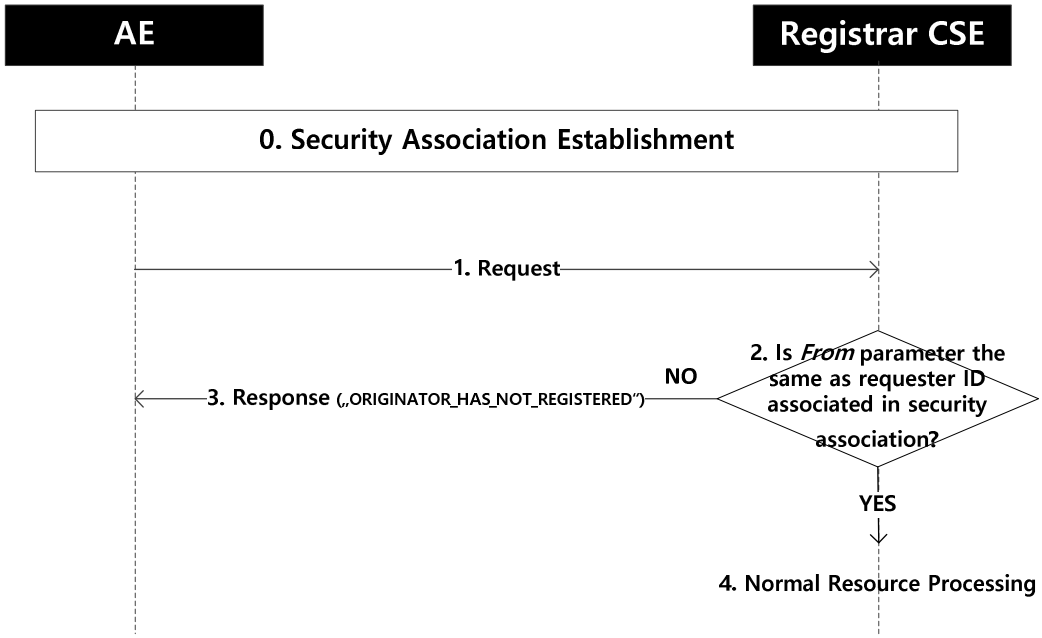


Figure 7.2.1-1: AE impersonation checking procedure

0. Security association establishment may be performed. Clause 6.1.2.2.1 describes the scenarios when security association establishment between an AE and CSE is mandatory, and describes the scenarios when security association establishment between an AE and CSE is recommended. The subsequent procedures shall be performed if a security association has been established.

1. The AE sends a request to Hosting CSE via its Registrar CSE as specified in ETSI TS 118 101 [1] (Hosting CSE is not represented on this figure and can either be the Registrar CSE or another CSE).

2. The Registrar CSE checks if the value in the ***From*** parameter is the same as the ID associated in security association:

3. If the values are not identical, then the Registrar CSE shall send a response with Response Status Code '4106' ("ORIGINATOR_HAS_NOT_REGISTERED").

4. If the values are identical, then the Registrar CSE shall perform the procedures specified in clause 8.2 of ETSI TS 118 101 [1]. Depending on the number of Transit CSEs, the Registrar CSE shall either process the request or forward it to the Hosting CSE or to another Transit CSE.

NOTE: This impersonation verification procedure is not applicable for CSE. This is because when a Transit CSE forwards a request to another CSE, the ***From*** parameter of the request is the identifier of the Originator which is different from the identifier of the Transit CSE.

7.2.2 Verification Using End-to-End Security of Primitives (ESPrim)

End-to-End Security of Primitives (ESPrim), clause 8.4, allows a Target (a Hosting CSE or AE) to authenticate the Originator of a request primitives that are processed by Transit CSEs. ESPrim also provides confidentiality and integrity protection of these request and response primitives. The primitives being protected are called the inner primitives. ESPrim encryption is applied to the inner primitives to form ESPrim Objects. Outer primitives are used to transport the ESPrim objects between the Originator and Target CSE or AE. The Originator's Registrar cannot view the encrypted inner primitive, and cannot verify that the ***From*** parameter of the inner primitive is correct. Instead, the Target is expected to verify that the ***From*** parameter of the inner primitive agrees with the authenticated identity of the Originator.

When the Target receives an ESPrim-protected request, the Target shall perform the following procedure.

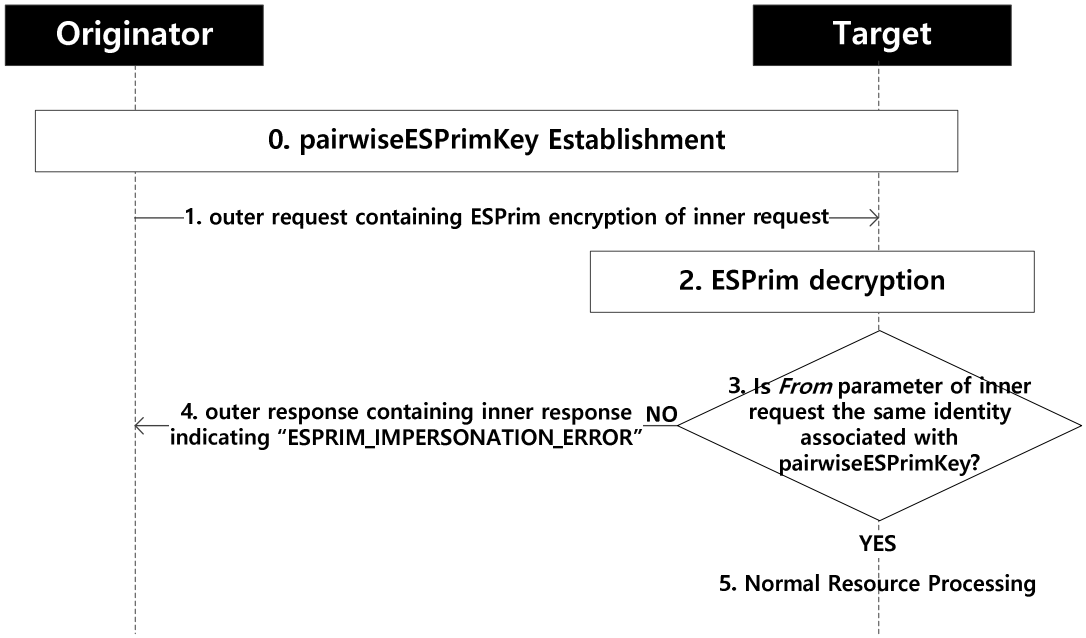


Figure 7.2.2-1: Impersonation checking procedure

0. The Target and Originator have previously established a symmetric pairwiseESPrimKey. The Target associates an identity with the symmetric pairwiseESPrimKey.

1. The Originator composes the inner request primitive, encrypts it using ESPrim to form an ESPrim Object, and sends it to the Target as described in clause 8.4.

NOTE: Regardless of whether ESPrim is applied, each Mcc "hop" is always protected using an SAEF, and each Mca "hop" is optionally protected using an SAEF; see clause 6.1.2.2.1.

2. The Target applies the procedures in clause 8.4 to decrypt the ESPrim Object and obtain the inner request primitive.

3. The Target checks if the value in the ***From*** parameter is the same as the ID associated with the pairwiseESPrimKey:

4. If the values are not identical, then the Target shall send a response with Response Status Code '4116' ("ESPRIM_IMPERSONATION_ERROR").

5. If the values are identical, then the Target shall record that the Originator has been authenticated, and performs procedures specified in clause 8.2 of ETSI TS 118 101 [1].

7.3 Dynamic Authorization

7.3.1 Purpose of the Dynamic Authorization

The Dynamic Authorization provides an interoperable framework for an Originator to be dynamically issued with temporary permissions providing the Originator with access to one or more resources on one or more CSEs.

Applicable use cases, requirements and proposals are discussed in oneM2M TR-0019 [i.15].

The present document specifies the exchanged Dynamic Authorization parameters and associated processing at the Originator and Hosting CSE. The transport of dynamic authorization parameters is specified in ETSI TS 118 101 [1] and ETSI TS 118 104 [4].

7.3.2 Dynamic Authorization Stage 2 Details

7.3.2.1 Dynamic Authorization Reference Model

The Dynamic Authorization reference model is shown in figure 7.3.2.1-1.

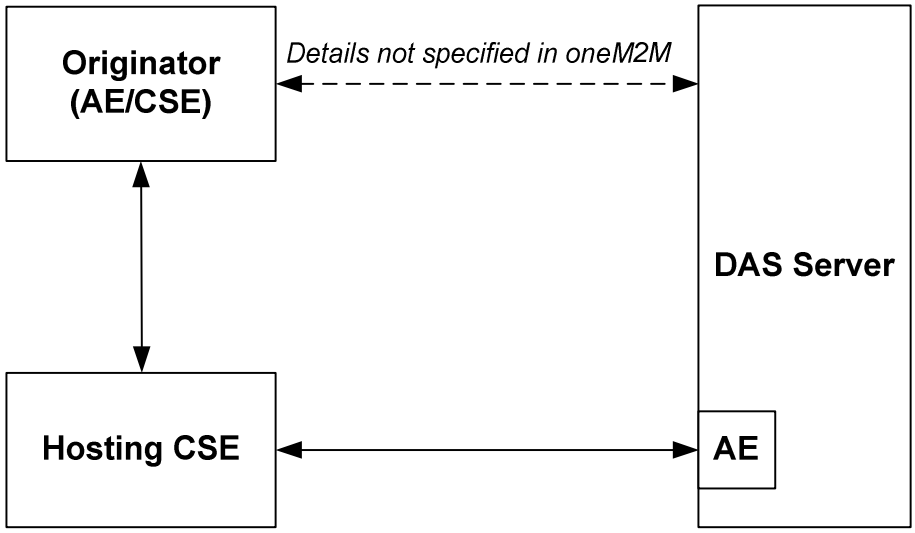


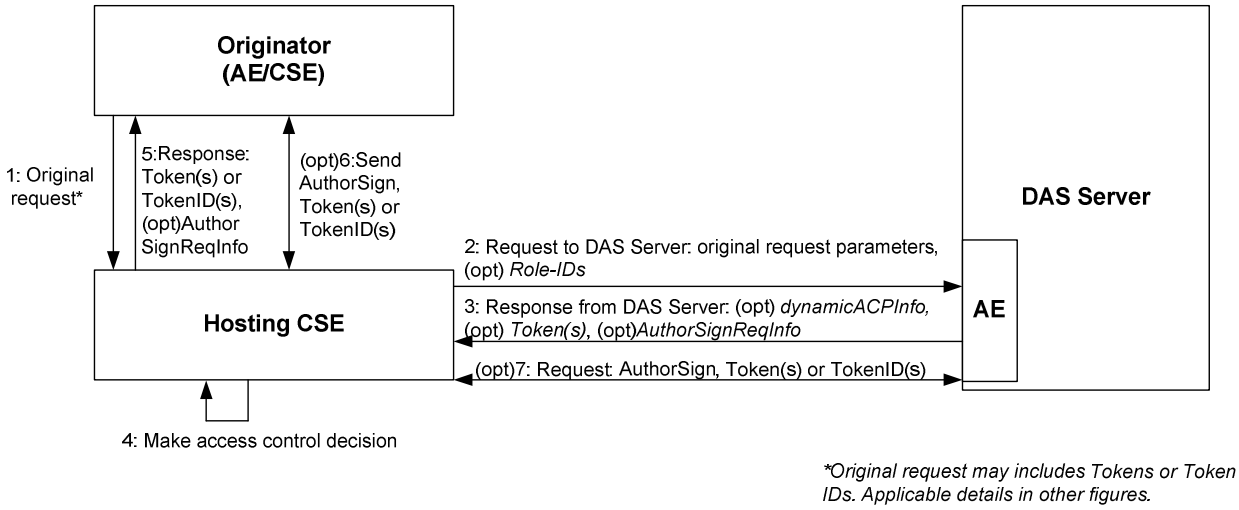
Figure 7.3.2.1-1: Dynamic Authorization reference model

The Dynamic Authorization reference model introduces the following systems and entities:

- Dynamic Authorization System (DAS): A system supporting dynamic authorization on behalf of resources owners. The present document does not describe the processing and exchange of messages within the Dynamic Authorization System. This system may reside either internally or externally within the service provider network.
- Dynamic Authorization System (DAS) Server: A server configured with policies for dynamic authorization, and provided with credentials for issuing Tokens. The DAS Server may include an AE for interaction with the oneM2M system.

The following Dynamic Authorization procedures are specified:

- **Direct Dynamic Authorization**, summarized in figure 7.3.2.1-2. In this procedure, Hosting CSE interacts with the DAS Server to obtain Dynamic Authorization. When AE, Hosting CSE and the DAS server support creating the Authorization Relationship Mapping Record, then steps 5-7 will be applied.



NOTE: Original request may include Tokens or Token IDs. Applicable details in other figures.

Figure 7.3.2.1-2: Direct Dynamic Authorization

- **Indirect Dynamic Authorization**, summarized in figure 7.3.2.1-3:

  - Steps 1 and 2: The Hosting CSE may provide the Originator with ***Token Request Information*** in the unsuccessful response.

  - Step 3: The Originator interacts with the DAS Server with the intention that the DAS Server issue *Tokens* authorizing the Originator, and the Originator is provided with the Token or a Token-ID. If the Originator is an AE, whose AE-ID-Stem is assigned by the registrar CSE, and both AE and DAS server support to create the Authorization Relationship Mapping Record, then the DAS Server shall request the AE to create the authorization relationship mapping record. The interaction is not described in the present document.

  - Step 4: If the DAS Server starts the process of AuthorRelMapRecord creation in step 3, then the AE shall create the AuthorRelMapRecord in the DAS Server.

  - Steps 5 to 8: The Originator provides the Hosting CSE with a *Token*, *Token-ID* to indicate that the Token is to be considered in the access decision. In the case of a token-ID, the Hosting CSE retrieves the corresponding Token via an AE of the DAS Server. These are then used in the access decision. If the AuthorRelMapRecord is created in step 4, then the Originator shall also indicate the related information to the Hosting CSE. The Hosting CSE may provide the Originator with a *Local-Token-ID* that may be used to identify the Token.

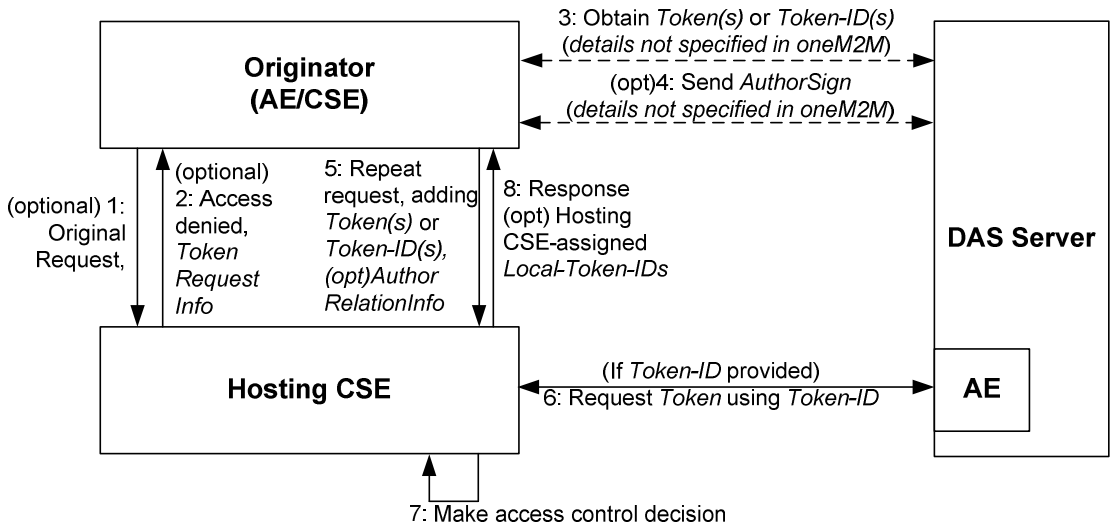


Figure 7.3.2.1-3: Indirect Dynamic Authorization

7.3.2.2 Direct Dynamic Authorization

The present document specifies the exchanged parameters and associated processing at the Hosting CSE. The transport of parameters is specified in clause 11.5.2, ETSI TS 118 101 [1].

The message flow for the Direct Dynamic Authorization is shown in figure 7.3.2.2-1, and described in the following text.

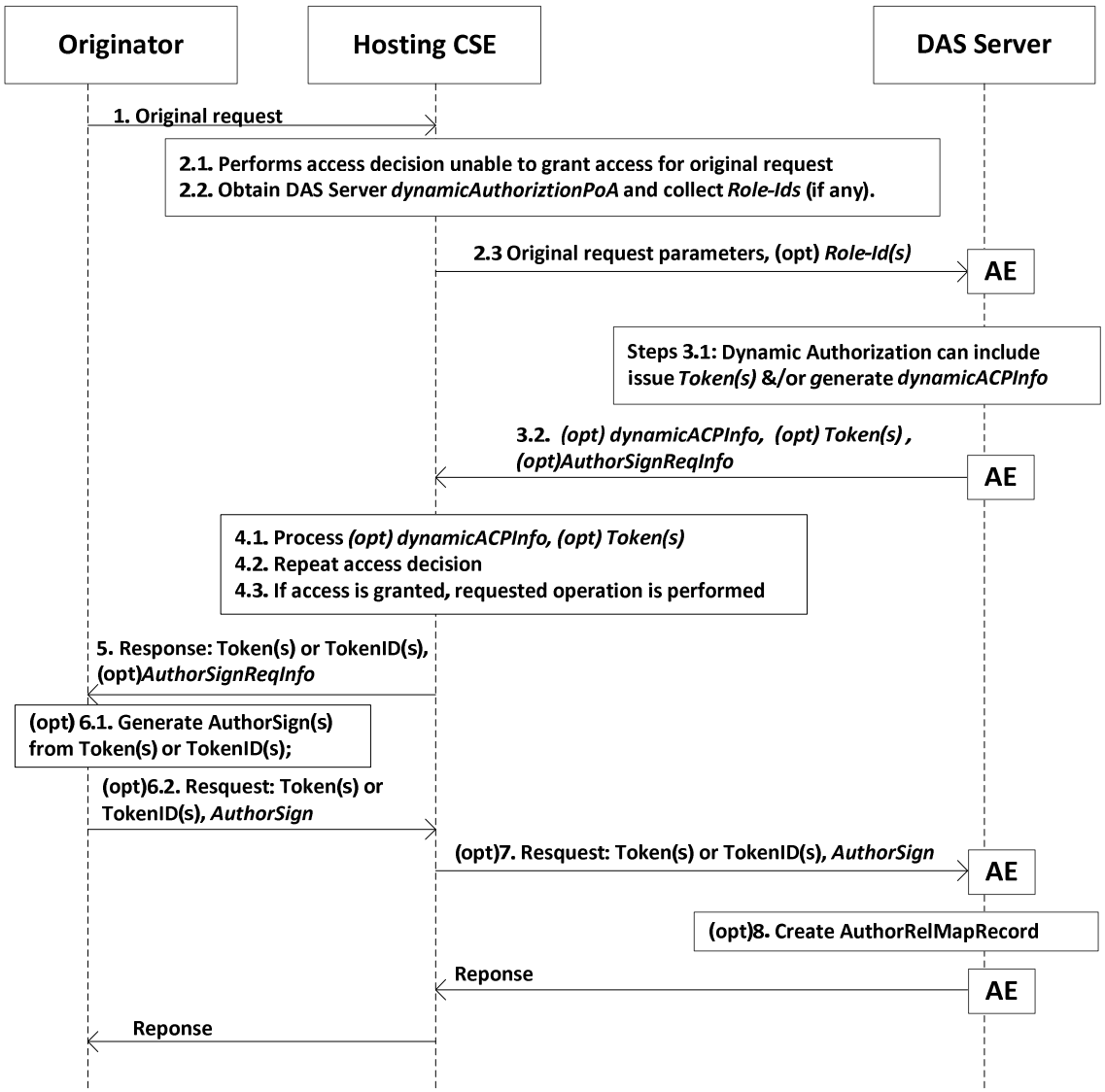


Figure 7.3.2.2-1: Message flow for Direct Dynamic Authorization

1. The Originator sends request (called the request from the Originator for this message flow) to the Hosting CSE. This request may include *Tokens* or *Token-IDs*; see the clause 7.3.2.3.

2. Initial Hosting CSE processing:

   2.1 If the request from the Originator includes ***Tokens*** or ***Token-IDs*** then these are processed as described in clause 7.3.2.3. The Hosting CSE evaluates the access decision algorithm, but is unable to grant access for the request from the Originator based on configured access control policies.

2.2 The Hosting HCSE determines the set of DAS Server with which Direct Dynamic Authorization may be performed:

2.2.1 The HCSE examines all *accessControlRules* for which request satisfies the *accessControlOperations* and *accessControlContexts* in the <*accessControlPolicy*> resources linked to the requested resource. The HCSE collects the set of all *Role-IDs* in the *accessControlOperators* of these *accessControlRules.* This *Role-IDs* are grouped according to the DAS Server AE-ID identified by the *Role-ID.*

NOTE 1: Regarding the Role-ID(s) parameter: The Originator would be granted access if a Token(s) is issued which associates the Originator with one or more of the Role-ID(s). Providing this list to the DAS Server allows the DAS Server to select a suitable set of one or more Role-ID(s) to associate with the Originator in Token(s), thereby authorizing the Originator to access the requested resources. The policies configured to the DAS Server would dictate which Role-ID(s) (if any) are included in Token(s) issued to the Originator.

2.2.2 The HCSE shall also collect the set of <*dynamicAuthorizationConsultation*> resources linked to the requested resource, and group these according to the DAS Server's dynamicAuthorizationPoA attribute of the <*dynamicAuthorizationConsultation*> resource.

2.3 The Hosting CSE selects a DAS Server (from the set determined in step 2.2) and sends a oneM2M request message containing the information described in table 7.3.2.2-1. The transport of parameters is specified in step 2.3, clause 11.5.2, ETSI TS 118 101 [1].

Table 7.3.2.2-1: Information sent from Hosting CSE to DAS Server during the Direct Dynamic Authorization

Parameter	Description	Mandatory/ Optional
Originator	Identifier of the Originator of the request received by the Receiver	M
Originator Resource Type	Type of resource targeted by originated request received by Receiver	M
Operation	Type of operation specified in originated request received by the Receiver	M
Originator IP Address	IP address of Originator of request received by Receiver	O
Originator Location	Location of Originator of request received by Receiver	O
Originator Role IDs	Role IDs of Originator of request received by Receiver	O
Request Timestamp	Timestamp when originated request was received by Receiver	O
Targeted Resource ID	Resource ID targeted by originated request received by Receiver	O
Proposed Privileges Lifetime	Proposed lifetime of authorization privileges requested by the Receiver	O
Role IDs From ACPs	The set of Dynamic Access Roles in the *accessControlDynAuthRole* parameters associated with the DAS Server AE-ID.	O
Token IDs	The set of token identifiers associated with the Originator	O
AuthorSignIndicator	An indicator included in the request received by Receiver to indicate the capability to sign for creating AuthorRelMapRecord when Originator is an AE. It is used in the case that the AE-ID-Stem is assigned by the Registrar CSE of the AE so that the AE-ID may change in a new registration (see clause 7.3.2.7.1). If the Hosting CSE does not support this parameter, then the Hosting CSE shall ignore it.	O

3. DAS Server processing:

3.1 The DAS Server processes the received parameters. The DAS Server may decide to provide *Token(s)* and/or *dynamicACPInfo* which will be used by the Hosting CSE to create a dynamic <*accessControlPolicy*> resource. The DAS Server applies the policies with which it is configured to decide on the appropriate actions.

NOTE 2: The details of this decision are specific to the Dynamic Authorization System being employed; these details are not visible to the oneM2M system, and are not addressed in the present document.

NOTE 3: The DAS Server that is contacted by the Hosting CSE can contact one or more additional DAS Servers to request additional tokens (e.g. tokens applicable to more sensitive or privileged resources). The DAS Server contacted by the Hosting CSE can then return a list of one or more Tokens to the Hosting CSE. The Issuer field inside the Token structure defined in clause 7.3.2.4 will indicate which DAS Server provided which Token. The messaging that takes place between DAS Servers is out of scope of oneM2M specifications.

The Token(s) (if any) shall conform to clause 7.3.2.4, with the following profile:

- The "holder" parameter shall contain the Originator's Absolute CSE-ID or AE-ID received from the HCSE, and may contain other CSE-IDs and AE-IDS.
- The "audience" parameter shall contain only the HCSEs CSE-ID.

The DAS Server shall apply an ESData protection option to the individual Tokens with the following requirements

- The DAS Server may encrypt the Token such that the Token can be decrypted by the Hosting CSE.
- The Hosting CSE shall be able to verify that the DAS Server issued the token.

The ESData processing results in an ESData envelope which is called the *ESData-protected Token* for the purposes of this message flow.

If the DAS Server decides to authorize the Hosting CSE to create a dynamic <*accessControlPolicy*> resource, then the DAS Server shall form a *dynamicACPInfo* parameter containing the following information are listed in table 7.3.2.2-2.

Table 7.3.2.2-2: Information included in the *dynamicACPInfo* parameter

Parameter	Description	Mandatory/Optional
Granted Privileges	List of granted privileges	O
Privileges Lifetime	Lifetime of granted privileges	O
Tokens	List of issued tokens	O

3.2 The DAS Server shall send the ESData-protected *Token(s)* (if any) and (optional) *dynamicACPInfo* parameter via the DAS Server AE to the Hosting CSE. The transport of parameters is specified in step 2.3, clause 11.5.2 of ETSI TS 118 101 [1]. If the DAS Server receives the *AuthorSignIndicator* from the Hosting CSE and the DAS server itself also supports to trigger creating the authorization relationship mapping record, then the DAS Server shall send an *AuthorSignReqInfo* to the Hosting CSE to request the AE to create the authorization relationship mapping record.

4. HCSE Processing:

4.1 The HCSE processes the ESData-protected *Token(s)* (if present) and *dynamicACPInfo* parameter (if present):

4.1.1 The HCSE shall perform the following verifications for each ESData-protected *Token*:

4.1.1.1 The HCSE shall apply ESData processing to the ESData-protected *Token* to extract the authenticated Token.

4.1.1.2 The HCSE shall perform the following verifications:

4.1.1.2.1 The "issuer" parameter in the Token shall exactly match the identity of the DAS Server.

4.1.1.2.2 The HCSE's CSE-ID shall match the CSE-ID in the "audience" parameter in the Token.

4.1.1.2.3 The "holder" parameter in the Token shall exactly matches the Absolute CSE-ID or AE-ID of the Originator from whom the request was received.

4.1.1.2.4 The HCSE shall verify that the Token has not expired, by comparing the current time to the "notAfter" parameter in the Token.

4.1.1.3 The HCSE shall cache the verified Token, and may later delete the verified Token when the Token expires (as defined in step 4.1.2.4). If the Hosting CSE receives an *AuthorSignReqInfo* from DAS Server AE, then the Hosting CSE shall make sure the Absolute AE-ID of the Originator shall be assigned to the *holder* attribute of the cached token.

4.1.2 If *dynamicACPInfo* is provided by the DAS Server, then the Hosting CSE shall create a dynamic <*accessControlPolicy*> resource matching the *dynamicACPInfo*.

4.2 The Hosting CSE repeats the access decision mechanism in clause 7.1.4.

4.3 If access is granted, then the Hosting CSE performs the operation requested in the request from the Originator, resulting in the Hosting CSE sending a request to the Originator.

5. The Hosting CSE shall send a response message containing the ESData-protected Token(s) (if present) or TokenID(s), and *AuthorSignReqInfo* if the Hosting CSE receives and supports the *AuthorSignReqInfo* from DAS Server AE. If the *AuthorSignReqInfo* is not included in the response, then the steps 6 to 8 will not be applied.

6. Originator processing:

6.1 If the Originator receives AuthorSignReqInfo, then the Originator shall generate AuthorSign(s) on Token(s) or TokenID(s) for each Token.

NOTE 4: AuthorSign is a signature generated using the certificate of the AE or a MIC generated using a symmetric key shared between the AE and Hosting CSE. How a symmetric key is distributed to the AE and DAS server is not specified in the present document.

NOTE 5: If the Originator includes the AuthorSignIndicator in step 1, but there is no AuthorSignReqInfo included in the response in step 5, then it indicates that the Hosting CSE or the DAS server does not support creating the authorization relationship mapping record.

6.2 The Originator sends the AuthorSign(s) with the corresponding Token(s) or TokenID(s) to the Hosting CSE.

7. The Hosting CSE forwards the parameters from the Originator to the DAS server AE.

8. DAS server AE shall create AuthorRelMapRecord(s) containing the following information listed in table 7.3.2.2-3 for each Token:

Table 7.3.2.2-3: Information included in the AuthorRelMapRecord

Parameter	Description	Mandatory/Optional
SubjectID	Absolute AE-ID of the AE	O
Token	The token issued for the AE	M
SignatureAuthorSign	Generated from Token or TokenID	M
ResourceID	The resource ID of the resource AE requests to access	O

7.3.2.3 Indirect Dynamic Authorization

The present document specifies the exchanged parameters and associated processing at the Originator and Hosting CSE. The transport of parameters is specified in clause 11.5.3 of ETSI TS 118 101 [1].

The message flow for Indirect Dynamic Authorization is shown in figure 7.3.2.3-1, and described in the following text.

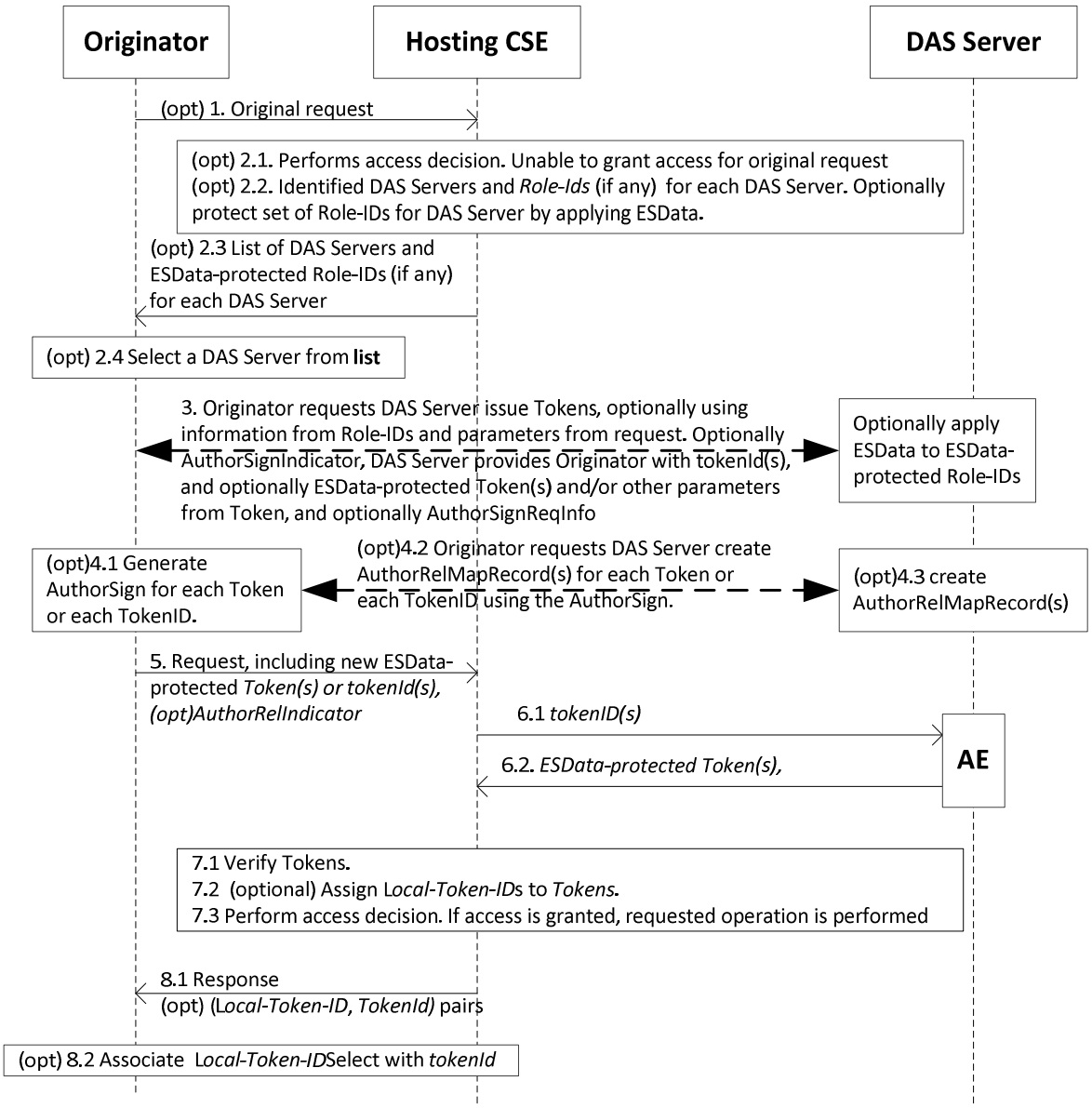


Figure 7.3.2.3-1: Message flow for Indirect Dynamic Authorization

1. (Optional) The Originator sends request to the Hosting CSE. The Originator includes an indication that the Originator is prepared to request Tokens from DAS Servers for this request. This request may include a combination of *Tokens*, *tokenIDs*, *Local-Token-IDs* but this message flow assumes that these do not provide sufficient permissions for accessing the requested resource.

2. (Optional) Initial Hosting CSE processing:

   2.1 Hosting CSE performs the access decision for the request from the Originator. This call flow assumes that the request from the Originator is denied as a result of the access decision. The Hosting CSE observes the indication that the Originator prepared to request Tokens from DAS Servers for this request.

   2.2 The Hosting CSE forms a list of DAS Server's and associated Role-ID(s) (if any) as described in step 2.2.1 of the Direct Dynamic Authorization in clause 7.3.2.2.

   For each DAS Server, then Hosting CSE may apply ESData to the set of Role-IDs for decryption by the DAS Server. For example, the ESData may encrypt the set of Role-IDs so they are not visible to the Originator.

   2.3 The Hosting CSE shall send an unsuccessful response to the Originator, including the list of DAS Servers and associated set of optionally-ESData-protected Role-IDs.

2.4 The Originator selects a DAS Server identified in the response.

3. The Originator shall interact with the DAS Server to request the issuance of a *Token*. The Originator can provide the optionally-ESData-protected set of Role-IDS to the DAS Server, and parameters from the original resource access request. If the Originator is an AE and the AE-ID-Stem is assigned by the Registrar CSE of the AE, and the Originator supports to create the authorization relationship mapping record, then the Originator shall provide the *AuthorSignIndicator* parameter in order to ask the DAS server to maintain the authorization relationship (see clause 7.3.2.7.2) in case the AE-ID of the Originator may change in a new registration. If the set of Role-IDS is protected using ESData, the DAS Server applies ESData to extract the set of Role-IDS. The DAS Server issues a Token(s) and provides the tokenID(s) and optionally the ESData-protected Token(s) to the Originator. The DAS Server can also provide the Originator with other parameters from the Token; for example, the time window in which the Token is valid. If the DAS Server receives the *AuthorSignIndicator* from the Originator, and the DAS server supports creating the authorization relationship mapping record, then the DAS server shall provide the Originator with an *AuthorSignReqInfo* to request the Originator to return *AuthorSign(s)* for each Token. This interaction is specific to the Dynamic Authorization System technology being used.

NOTE 1: The DAS Server that is contacted by the Hosting CSE can contact one or more additional DAS Servers to request additional tokens (e.g. tokens applicable to more sensitive or privileged resources). The DAS Server contacted by the Hosting CSE can then return a list of one or more Tokens to the Hosting CSE. The Issuer field inside the Token structure defined in clause 7.3.2.4 will indicate which DAS Server provided which Token. The messaging that takes place between DAS Servers is out of scope of oneM2M specifications.

4. If the Originator receives an *AuthorSignReqInfo* from DAS server, then the Originator shall return the AuthorSign(s) to DAS server:

4.1 The Originator generates *AuthorSign(s)* on Token(s) or TokenID(s) for each Token.

NOTE 2: AuthorSign are a signature generated using the certificate of the AE or a MIC generated using a symmetric key shared between the AE and DAS server. How a symmetric key is distributed to AE and DAS server is not specified in the present document.

4.2 The Originator sends the AuthorSign(s) to DAS server with the corresponding Token(s) or TokenID(s).

4.3 The DAS server shall create AuthorRelMapRecord(s) containing the information listed in table 7.3.2.2-3 for each Token.

5. For request that the Originator wishes to have authorized using an issued Token, the Originator shall add ESData-protected Token provided by the DAS Server or *tokenID* (if no ESData-protected Token was provided) if the corresponding ESData-protected Token(s) was not provided by the DAS Server. In particular, if the request at step 1 was unsuccessful at step 2.3, then the Originator may repeat the request with new *Token(s)* and/or *tokenID(s)*. A token may be used in multiple request. If step 4 is performed, then the request shall contain the *AuthorRelIndicator* to indicate to the Hosting CSE that the relationship between the AE and the Token(s) are maintained in the DAS server.

The Originator shall send the request to the Hosting CSE.

6. (Optional) If the request includes *tokenID(s)*, then for each *tokenID* the Hosting CSE identifies the corresponding DAS Server AE from which to request the corresponding Token:

6.1 The Hosting CSE sends the tokenID(s) to the DAS Server via a DAS Server AE.

6.2 The DAS Server shall return the corresponding valid ESData-protected Token(s) to the Hosting CSE via the DAS Server AE.

7. Hosting CSE Processing:

7.1 Token Processing:

7.1.1 The Hosting CSE shall apply ESData to the ESData-protected Token(s), either provided in the request or retrieved from the DAS Server, to extract the authenticated Token(s).

7.1.2 If a Local-Token-ID was provided in the request, then the Hosting CSE attempts to retrieve the cached token.

7.1.3 The HCSE shall perform the following verifications for each authenticated and cached token associated with the request:

- The HCSE's CSE-ID shall match one of the Absolute CSE-IDs (optionally including wildcards) in the "audience" parameter in the Token.
- The "holder" parameter in the Token shall exactly match the Absolute CSE-ID or AE-ID of the Originator from whom the request was received.
- The HCSE shall verify that the Token is currently valid and not expired, by comparing the current time to the "notBefore" and "notAfter" parameter in the Token. If a cached Token has expired, then the Token may be removed from the cache.

7.1.4 If any identified Token could not be retrieved in steps 6 or 7.1.2, or if any ESData-protected Token-ID failed verification at step 7.1.1, or if any Token failed the verification at step 7.1.3, then the Hosting CSE shall respond with an error.

7.1.5 The Hosting CSE may cache any new Token(s). If the Hosting CSE receives the *AuthorRelIndicator* in step 5, then the Hosting CSE shall make sure the Absolute AE-ID of the Originator is assigned to the *holder* attribute of the cached token.

7.2 The Hosting CSE may assign *Local-Token-ID(s)* to cached Token(s).

7.3 The Hosting CSE shall perform the access decision as described in clause 7.1.4, including the information in the Token(s) identified in the request. If access is granted, then the requested operation shall be performed.

8. Response:

8.1 The Hosting CSE sends a response to the Originator. For each new *Local-Token-ID*(s) that has been assigned, the Hosting CSE provides the *Local-Token-ID* and corresponding *tokenID* in the response parameters.

8.2 The Originator associates the *Local-Token-ID* with *tokenID*. In subsequent requests, the Originator may use the *Local-Token-ID* instead of the *Token* or *tokenID*.

7.3.2.4 Token Structure

A token is used to carry authorization information that can be roles assigned to the token holder or access control policies applicable to the token holder. The structure of token is shown in figure 7.3.2.4-1, it contains the following data fields:

- version: version of the token.
- tokenID: unique ID of the token.
- holder: ID of the token holder.
- issuer: ID of the token issuer.
- notBefore: token valid from this time.
- notAfter: token expired after this time.
- tokenName: optional, human readable name of the token.
- audience: optional, list of CSE_IDs of the CSEs expected to accept the token.
- permissions: permissions associated with the token. Its format is specified in clause 9.6.39 of ETSI TS 118 101 [1].
- extension: used for store other information, e.g. application-specific information.

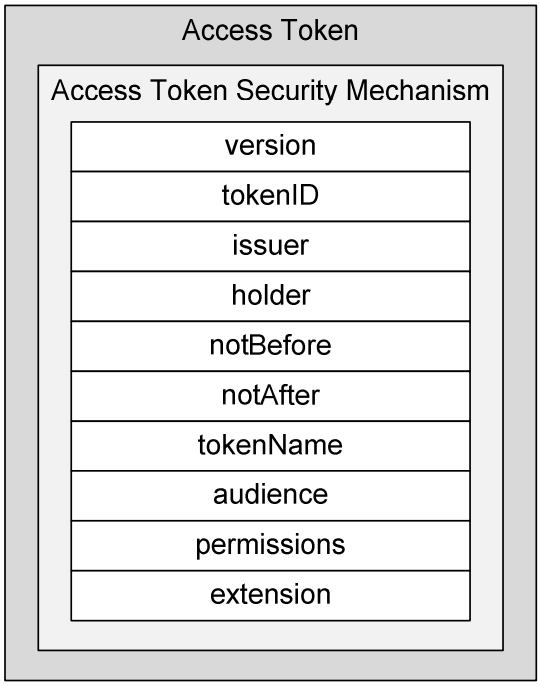


Figure 7.3.2.4-1: Structure of token

A token shall be protected by the ESData security mechanism. A token shall be signed, encrypted or signed and encrypted.

7.3.2.5 Token Evaluation

The generic process of evaluating a token can be described as follows:

1) Token security validation: Depending on the security mechanism used by a token, the validation may be:

   - Verifying signed token;
   - Decrypting encrypted token; or
   - Decrypting and verifying signed and encrypted token.

   After passing the token security validation, the plain text of the token can be used for further validation.

2) Token content validation: Depending on the content contained in the token, the validation may check:

   - If the identity of the Originator equal to the token holder specified in the *holder* data field.
   - If the token issuer specified in the *issuer* data field is valid.
   - If this token is not expired according to the *notBefore* and *notAfter* data fields.
   - If the identifier of the Hosting CSE is in the CSE-ID list specified by the *audience* data field (in case the *audience* data field is not empty).

   After passing the token content validation, the permissions associated to this token shall be used for access control.

3) Token permissions evaluation: Checking the permission element in the permission list one by one until the access request is permitted by one of the permissions or end of the list. For each permission in the list of the permissions the evaluation shall be done as follows:

   - Checking *resourceIDs* element. If it is present, then the authorization information described in *privileges* and/or *roleIDs* elements shall apply only to the resources specified by this element. If the *privileges* element is present, then this element shall be present.

- If the *privileges* element is present, the access control rules held in this element shall be used as applicable access control policy in the current access control decision making process.
- If the *roleIDs* element is present, the Role-IDs held in this element shall be used as valid roles in the current access control decision making process.

7.3.2.6 oneM2M JSON Web Tokens (JWTs)

7.3.2.6.1 Introduction to oneM2M JWTs

oneM2M specifies a JSON Web Tokens (JWTs) representation (IETF RFC 7519 [53]) for Tokens used in oneM2M. A JWT compliant with the present clause is called a *oneM2M JWT*.

Background: A JWT uses either the JSON Web Signature (JWS) Compact Representation, or JSON Web Encryption (JWE) Compact Representation, specified in IETF RFC 7515 [51] and IETF RFC 7519 [53]. The JWT specification IETF RFC 7519 [53] also defines an unsecured JWT which is a JWS using the "alg" Header Parameter value "none" and with the empty string for its JWS Signature value.

The JWT specification defines a JSON element which is the structure of the payload of the JWS or JWE when used as a JWT. This payload comprises a set of JWT claims, with IETF RFC 7519 [53] standardizing an initial set of JWT claim names. IANA maintains a registry of JWT claim names [i.18].

7.3.2.6.2 oneM2M JWT Profile

oneM2M JWT Claims: Table 7.3.2.6.2-1 provides the mapping from the JWT claim names, in a oneM2M JWT, to the elements of the m2m:tokenClaimSet complex data type described in ETSI TS 118 104 [4]. Where available, JWT claim names registered with IANA [i.18] have been used. ETSI TS 118 104 [4] specifies which elements are mandatory and which elements are optional.

Table 7.3.2.6.2-1: The oneM2M JWT claim set and mapping to elements of m2m:tokenClaimSet

Token Claimset Object Element Path	Token Claimset Object Element Short Name	oneM2M JWT claim name	Where is this JWT claim name defined?	Additional details for mapping from Token Claimset Object values to JWT Claim values
version	***tkvr***	"tkvr"	ETSI TS 118 104 [4] short names	Values shall be identical
tokenID	***tkid***	"jti"	IETF RFC 7519 [53]	Values shall be identical
issuer	***tkis***	"iss"	IETF RFC 7519 [53]	Values shall be identical
holder	***tkhd***	"azp"	OpenID Connect Core 1.0 [54]	Values shall be identical
notBefore	***tknb***	"nbf"	IETF RFC 7519 [53]	Token Claimset Object element "notBefore" is in ISO8601 [79] "Basic Format", see ETSI TS 118 104 [4]. This element shall be mapped to JWT Claim "nbf" which uses NumericDate format [53].
notAfter	***tkna***	"exp"	IETF RFC 7519 [53]	Token Claimset Object element "notAfter" is in ISO8601 [79] "Basic Format", see ETSI TS 118 104 [4]. This element shall be mapped to JWT Claim "exp" which uses NumericDate format [53].
tokenName	***tknm***	"tknm"	ETSI TS 118 104 [4] short names	Values shall be identical
audience	***tkau***	"aud"	IETF RFC 7519 [53]	Token Claimset Object element "audience" is a list of m2m:ID. This list shall be mapped to JWT Claim "aud" comprising an array of case-sensitive strings, each containing a StringOrURI value [53].
permissions	***tkps***	"tkps"	ETSI TS 118 104 [4] short names	Values shall be identical
extension	***tkex***	"tkex"	ETSI TS 118 104 [4] short names	Values shall be identical

oneM2M JWT Security Profile: The JWS Compact Representation and JWE Compact Representation are both supported by ESData (see clause 8.5.3). A oneM2M JWT may use any ESData security class: Encryption-only, Signature-only or Nested-Sign-then-encrypt. A oneM2M JWT may use any algorithm supported by ESData for the JWS Compact Representation and JWE Compact Representation.

A oneM2M JWT may be an unsecured JWT, in which case the oneM2M JWT is considered to use the unsecured ESData security class.

IETF RFC 7519 [53] discusses security considerations of JWTs, and operators of Token Issuers (Dynamic Authorization Servers and Authorization Authorities) should consult that text when deciding on ESData security class and algorithms.

JOSE header parameters of oneM2M JWTs: When the Encryption-only ESData security class is used, then:

- The JOSE header of the JWE shall include the "typ" parameters set to "JWT".
- The JOSE header of the JWE shall not include the "cty" parameter.

When the Signature -only ESData security class is used, then:

- The JOSE header of the JWS shall include the "typ" parameters set to "JWT".
- The JOSE header of the JWS shall not include the "cty" parameter.

When the Nested-Sign-then-encrypt ESData security class is used, then the JWT claims are the payload of a JWS, and the JWS becomes the payload of a JWE. In this case:

- The JOSE header of both the JWS and the JWE shall include the "typ" header parameters set to "JWT".

- The JOSE header of the JWE shall include the "cty" parameter set to be "JWT", to indicate that a Nested JWT is carried in this JWT.
- The JOSE header of the JWS shall not include the "cty" parameter.

7.3.2.6.3 oneM2M JWT Procedures

Configuring CSEs for verifying Tokens from a Token Issuer: In order for a CSE to verify oneM2M JWTs issued by a particular Token Issuer, the CSE shall be provided with the following information in a secure manner:

- The combinations of ESData Security classes and algorithms permitted by the Token Issuer.
- Credentials for verifying Tokens conforming to those ESData Security classes and algorithms, noting that no credentials are needed for verifying tokens using the unsecured ESData Security class.

The present document does not specify mechanisms for providing this information to the CSE. The present document does not define data structures for storing this information on the CSE. The security level to apply on each particular CSE has to be derived from application specific risk assessment.

Creating a oneM2M JWT: When a Token Issuer is triggered to create a token, then the Token issuer shall perform the following steps:

1) The Token Issuer shall form a Token Claimset Object compliant with data type m2m:tokenClaimSet, with the permission element using the JSON serialization.
2) The Token Issuer shall create the corresponding oneM2M JWT claim set using the mapping in table 7.3.2.6.2-1.
3) The Token Issuer shall select an ESData Security Class, algorithms and corresponding credentials. This step may also be performed before step 1) or between steps 1) and 2).
4) The Token Issuer shall create oneM2M JWT using the oneM2M JWT claims, ESData Security Class, algorithms and corresponding credentials. This step uses the process described for JWTs in IETF RFC 7519 [53].

The resulting oneM2M JWT has data type m2m:dynAuthJWT.

Validating a oneM2M JWT: When a CSE receives a oneM2M JWT for use in an access decision, then the CSE shall perform the following steps:

1) The CSE shall validate that the oneM2M JWT conforms to the m2m:dynAuthJWT data type.
2) The CSE shall validate the security of the oneM2M JWT as described in clause 7.3.2.5, using the JWT-specific details in IETF RFC 7519 [53] and configured credentials (if required). A CSE shall discard a oneM2M JWT which uses an ESData Security class or algorithms which are not permitted by the Token Issuer.
3) The CSE shall create a Token Claimset Object from the oneM2M JWT claim set by reversing the mapping in table 7.3.2.6.2-1.
4) The CSE shall validate the Token Claimset Object as described in clause 7.3.2.5.

The Token Claimset Object permissions element can now be processed as described in clause 7.3.2.5.

7.3.2.7 AE Authorization Relationship Update

7.3.2.7.1 AE Direct Authorization Relationship Update

This clause specifies the exchanged parameters and associated processing at AE and Hosting CSE.

The message flow for the Direct Authorization Relationship Update is shown in figure 7.3.2.7-1, which is described in the following text.

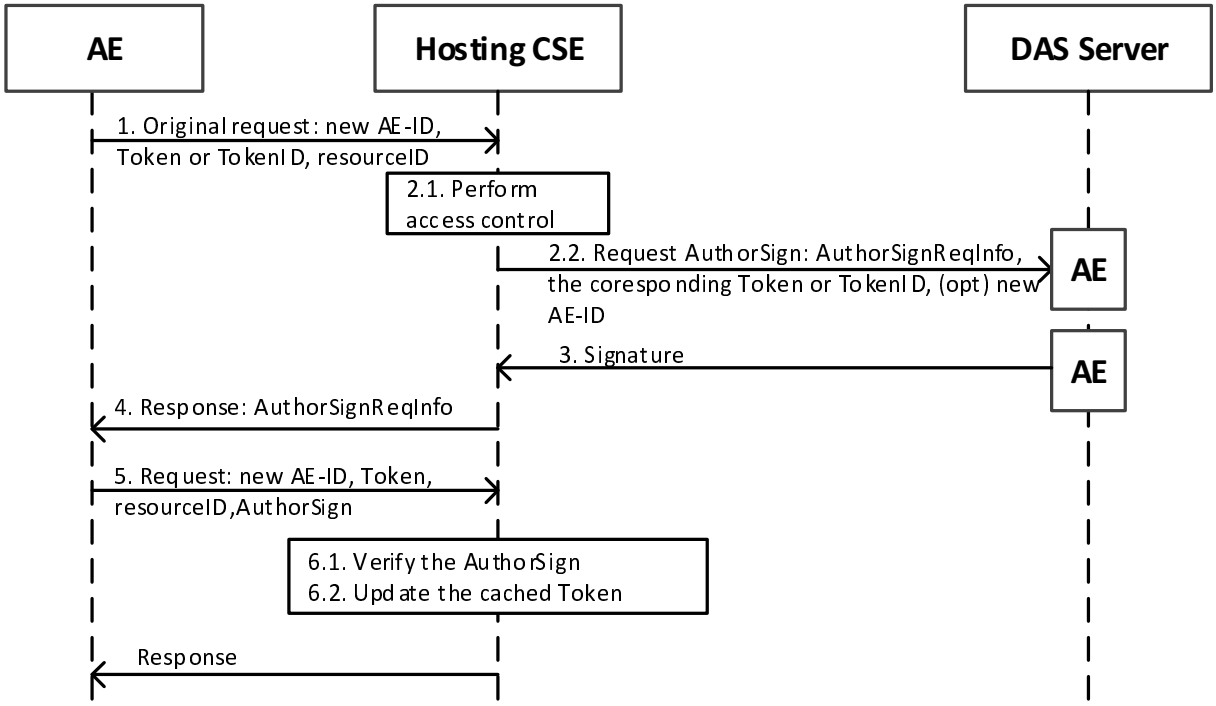


Figure 7.3.2.7-1: AE Direct Authorization Relationship Update

1. An AE sends a resource access request message to a Hosting CSE, which carries the new AE-ID, and the Token or the TokenID issued for it.

2. The Hosting CSE shall verify the Token or TokenID:

   2.1 The Hosting CSE shall verify whether this Token or the Token identified by this TokenID is valid, in the present document one way is provided to verify the Token, but there is no limitation on how to verify the Token. the Hosting CSE can search whether there is a cached Token which has the same TokenID as the TokenID received from the AE. If the *holder* attribute of the cached Token is not equal to the AE-ID of the originator, then the Hosting CSE performs the following steps to verify whether the AE has the possession of the Token.

   2.2 The Hosting CSE sends a request message to DAS Server AE to get the value of AuthorSign for this Token, which containing the information: *AuthorSignReqInfo*, the corresponding Token or TokenID received from the AE.

3. The DAS Server AE examines its AuthorRelMapRecord list to find if there is the record whose *Token* parameter or TokenID of the *Token* parameter is equal to the Token or TokenID in the request message. If a record exists, then the DAS Server AE returns the *Signature* parameter to the Hosting CSE.

4. The Hosting CSE rejects the request to access the resource, including an *AuthorSignReqInfo* in the response message to indicate AE to return the *AuthorSign* for this Token.

5. The AE sends the resource access request message again including the information: *AuthorSign*, *resourceID* and Token.

6. After receiving the *AuthorSign*, the Hosting CSE shall check whether the *AuthorSign* is equal to the value of *Signature* returned from DAS Server AE. If the signatures are identical, then the Hosting CSE shall update the value of the *holder* attribute of the cached Token on the Hosting CSE to the new AE-ID.

If DAS Server AE does not return a Signature value in step 3 or the result of the comparison in step 6 determines that the signatures are not identical, then the Hosting CSE shall refuse the request with no further process.

7.3.2.7.2 AE Indirect Authorization Relationship Update

This clause specifies the exchanged parameters and associated processing at AE and Hosting CSE.

The message flow for the Indirect Authorization Relationship Update is shown in figure 7.3.2.7-1, which is described in the following text.

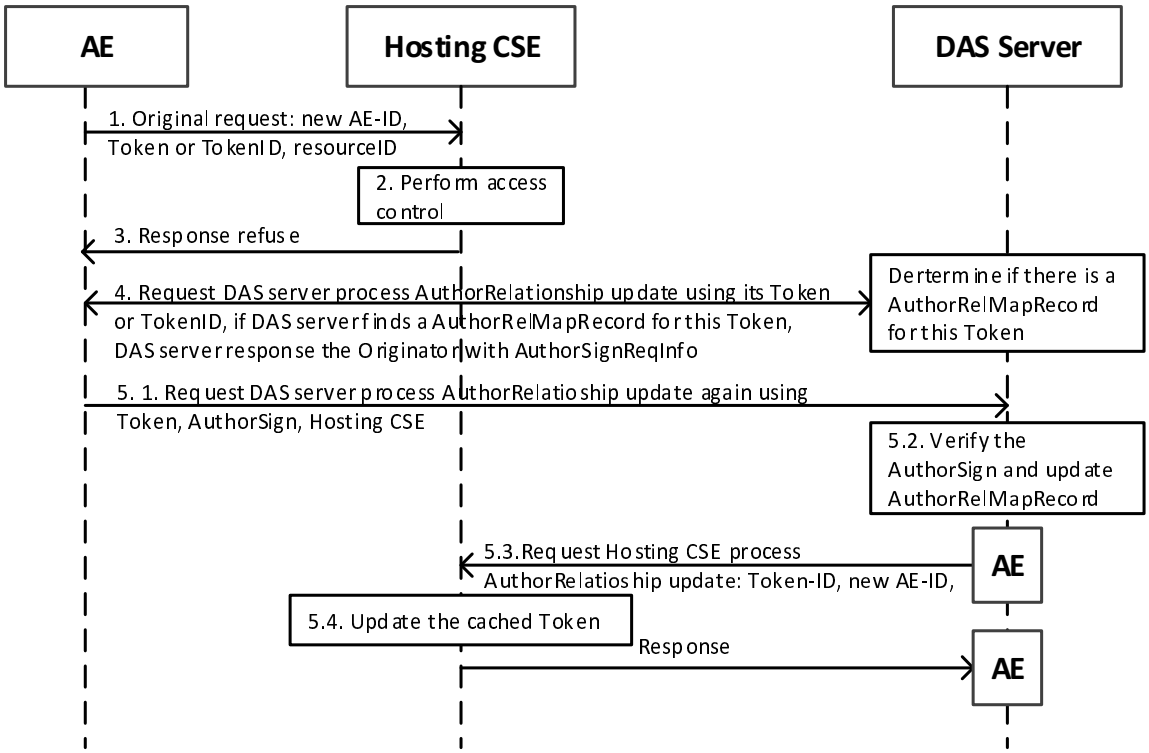


Figure 7.3.2.7-2: AE Indirect Authorization Relationship Update

1. The AE sends a resource access request message to the Hosting CSE, which carries the new AE-ID, and the Token or the TokenID issued for it.

2. The Hosting CSE verifies the Token or TokenID.

3. If the *holder* attribute of the cached Token is not equal to the AE-ID of the originator, then the Hosting CSE refuses the request to access the resource.

4. The AE requests the DAS Server to update the authorization relationship using the Token or TokenID, and the DAS server shall search if there is an AuthorRelMapRecord whose value of the *Token* parameter or TokenID of the *Token* parameter is the same with the Token or TokenID received from AE. If the result is ok, then the DAS server shall return an *AuthorSignReqInfo* to AE to request the *AuthorSign* of the Token.

5. The AE provides the *AuthorSign* to prove the possession of the Token:

   5.1 The AE sends the update request containing Hosting CSE ID, *AuthorSign*, resourceID and Token.

   5.2 After receiving the *AuthorSign*, the DAS Server shall check if this *AuthorSign* is equal to the value of *Signature* parameter in the AuthorRelMapRecord corresponding to this Token.

   5.3 If the check result in step 5.2 is true, then the DAS Server AE shall send a request message to the Hosting CSE to update the authorization relationship.

   5.4 The Hosting CSE updates the value of the *holder* attribute of the cached Token locally stored to the new AE-ID.

7.4 Role Based Access Control

7.4.1 Role Based Access Control Architecture

Figure 7.4.1-1 provides a high level overview of the role based access control architecture in the oneM2M System. The entities related to role issuance and role based access control are described as follows:

- Authorization Authority: It is responsible for assigning roles to Originators through creating <*role*> and/or <*token*> resources in Role and/or Token Repositories.
- Role Repository: It is a CSE that is responsible for storing <role> resources.
- Token Repository: It is a CSE that is responsible for storing <*token*> resources.

NOTE: The arrows shown in figure 7.4.1-1 are the logical relations among the entities and not the registration relations that form the real data paths.

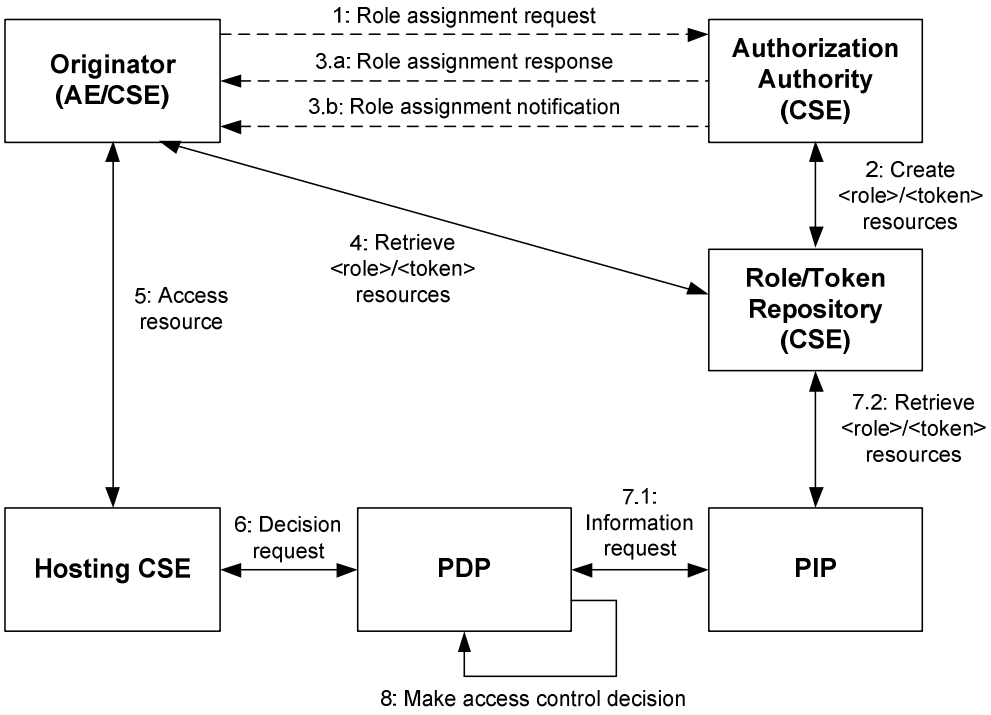


Figure 7.4.1-1: Role based access control architecture

The generic procedure of this architecture is described as follows:

1. An Originator may apply for a role from an Authorization Authority. This step may not exist in some situations, e.g. the Authorization Authority directly assigns a role to an Originator.

   This step is not specified in the present document.

2. The Authorization Authority shall check if the applied privilege can be assigned to the Originator. If it is permitted, the Authorization Authority need to create a <*role*> resource that specifies the role assignment in a role repository, or issue a token that contains the assigned role to the Originator. The issued token may be stored in a <*token*> resource in a token repository.

3. The Authorization Authority informs the Originator on the result of a role assignment. The returned information may contain role ID, token ID, token or the information about the created <role> or <token> resources. There are two cases to be considered.

   a) In case the Originator sends a role assignment request to the Authorization Authority, the Authorization Authority returns the result of the role assignment via a role assignment response.

b) In case the Authorization Authority directly assigns a role to an Originator, the Authorization Authority informs on the result of a role assignment via a role assignment notification.

This step is not specified in the present document.

4. The Originator may retrieve the assigned roles and/or tokens from a Role and/or Token Repositories using the information provided by an Authorization Authority in order to get detailed information about the assigned roles and/or tokens.

5. The Originator sends an access request to the target resource in the Hosting CSE. The request may contain the role information that may be the role IDs, tokens or token IDs.

6. The Hosting CSE may send an access control decision request to a PDP.

7. The PDP may need to retrieve the Originator's role assignment information according to the Role-IDs and/or Token-IDs from the Role and/or Token Repositories.

8. The PDP verifies the Originator's roles and/or tokens, and then makes an access control decision according to the access control policies and roles.

7.4.2 Role Issuing Procedure

7.4.2.1 Introduction

There are two ways to assign roles to an Originator:

1) One way is to create a <*role*> resource that describes what role is assigned to the originator. The <*role*> resources are stored in role repositories from which the AEs and CSEs can retrieve <*role*> resources in order to get an Originator's role assignment information.

2) Another way is to issue a token that describes what role is assigned to the token holder (i.e. the Originator). The issued token may also be stored in a <*token*> resource in a token repository from which the issued token can be retrieved.

A <*role*> resource may also point to a <*token*> resource in which the token that holds the assigned role is stored through the *tokenLink* attribute.

7.4.2.2 Role Assignment Procedure

The general procedure of assigning a role to an Originator is shown in figure 7.4.2.2-1 and described as follows:

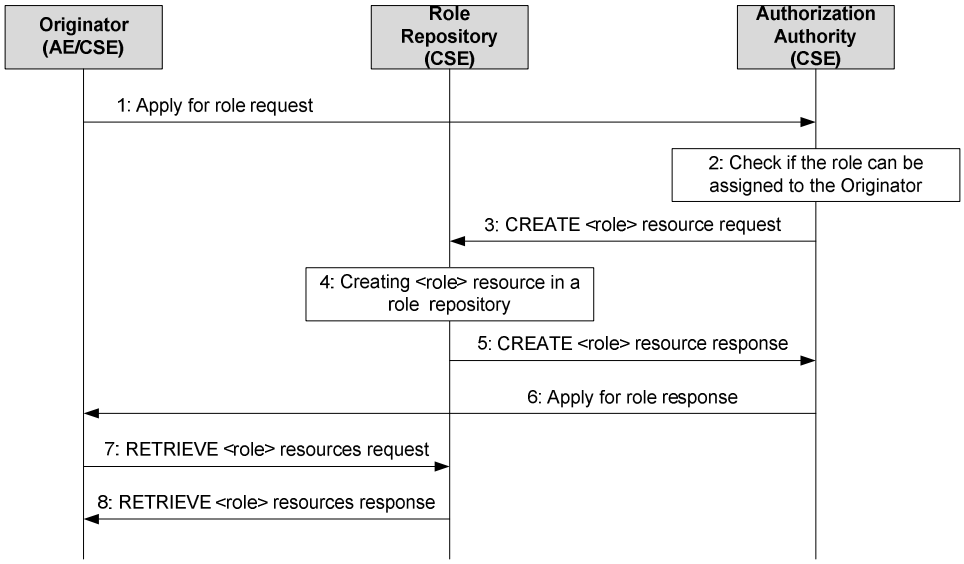


Figure 7.4.2.2-1: Procedure of role assignment

The procedure of role assignment is:

1. The Originator may send role assignment request to the Authorization Authority. The specification about this step is however out of scope of the present document.
2. The Authorization Authority shall check if the applied role can be assigned to the Originator. After passing the privilege authorization check, the Authorization Authority shall assign the role to the Originator.
3. The Authorization Authority shall send a <role> resource creation request to the Role Repository.
4. The Role Repository shall create a <role> resource according to the creation request.
5. The Role Repository shall return the result of <role> resource creation back to the Authorization Authority.
6. The Authorization Authority shall return the result of the role assignment back to the Originator. The specification about this step is however out of scope of the present document.
7. The Originator may send <role> resource retrieve request to the Role Repository in order to get the role assignment information.
8. The Role Repository shall return retrieved content of the <role> resources back to the Originator.

7.4.2.3 Issuing Token Associated with Role

The general procedure of issuing a role token (a token associated with assigned role) to an Originator is shown in figure 7.4.2.3-1 and described as follows:

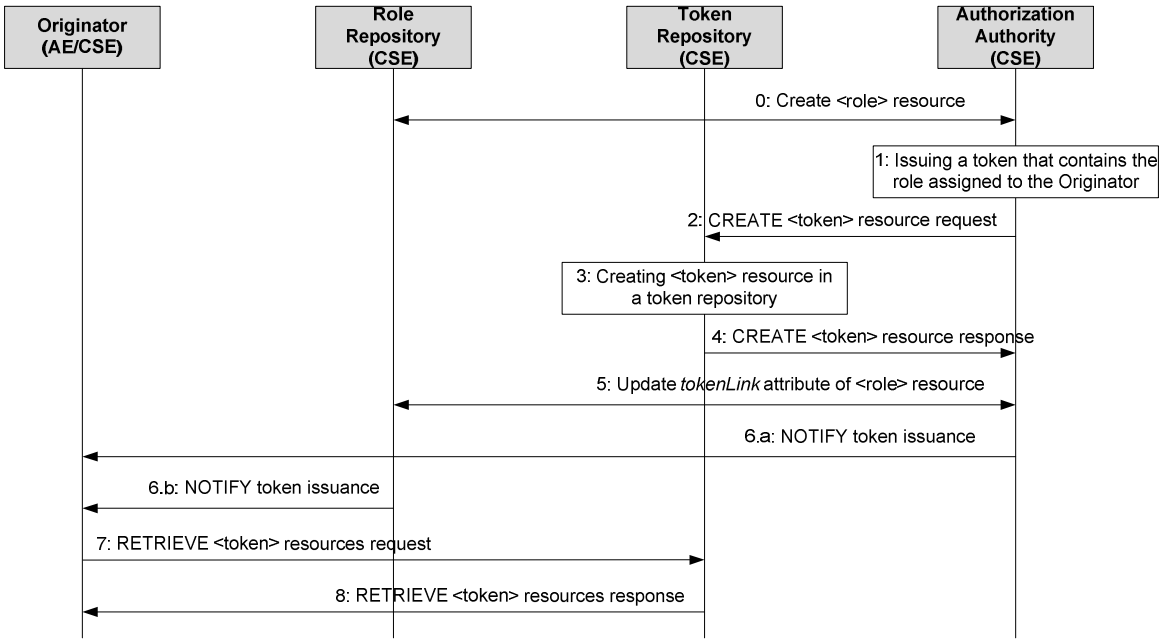


Figure 7.4.2.3-1: Procedure of role token issuance

The procedure of role token issuance is:

0. The Authorization Authority creates a <*role*> resource in a Role Repository for a role assignment.
1. The Authorization Authority issues a token that contains the role assigned to the Originator.
2. The Authorization Authority shall send a <*token*> resource creation request to a Token Repository.
3. The Token Repository shall create a <*token*> resource according to the creation request.
4. The Token Repository shall return the result of <*token*> resource creation back to the Authorization Authority.
5. The Authorization Authority shall update the *tokenLink* attribute of the <*role*> resource with the address of the <*token*> resource.

6. The token issuance information shall be passed to the Originator. There are two ways to deal with the notification:

   a) The Authorization Authority uses the NOTIFY operation inform the Originator.

   b) The Token Repository NOTIFY the Originator according to the subscription made by the Originator to the <role> resource.

7. The Originator may send a *<token>* resource retrieve request to the Token Repository in order to get the token issuance information.

8. The Token Repository shall return the retrieved content of the *<token>* resource back to the Originator.

7.4.3 Role Based Access Control Procedure

The general procedure of using a role in an authorization process is shown in figure 7.4.3-1 and described as follows:

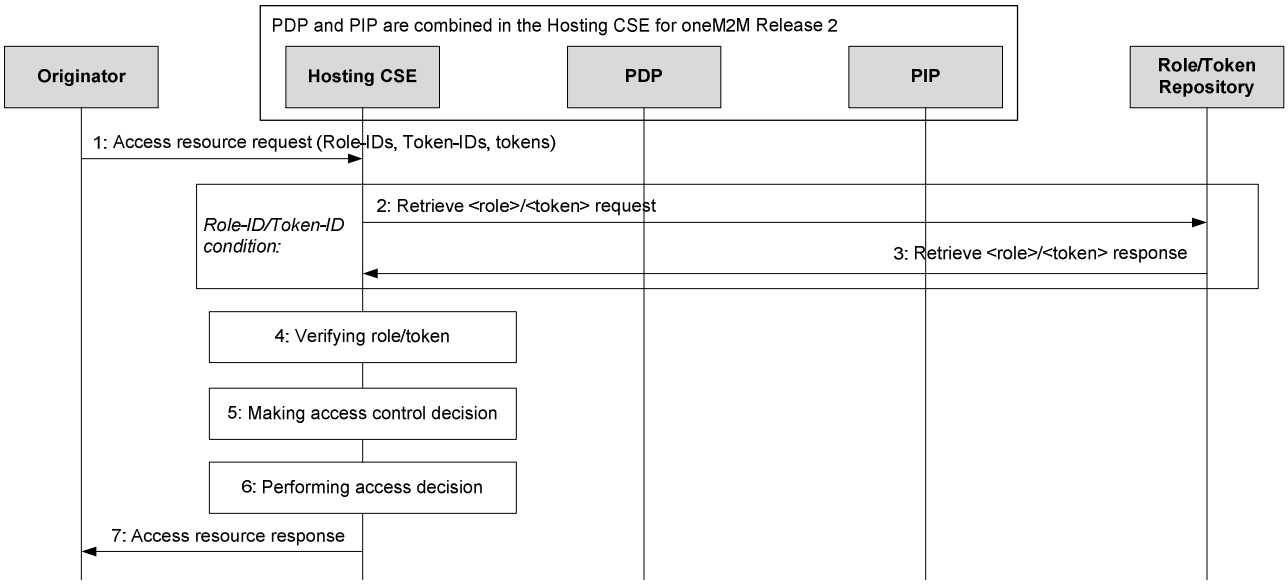


Figure 7.4.3-1: Role based access control procedure

1. The Originator shall include the applicable Role-IDs, Token-IDs or tokens into the request sent to the Hosting CSE.

2. In case of Role-IDs or Token-IDs, the Hosting CSE (acting the role of PIP) shall send a <role> or <token> resource retrieve request to the Role or Token Repository.

3. The Role Repository or Token Repository shall return the attributes of <role> or <token> resource back to the Hosting CSE.

4. The Hosting CSE (acting in the role of PDP) shall verify the received roles and/or tokens, the verification shall include: if a role/token is issued by a valid Authorization Authority, if holder of the role/token is equal to the Originator, and if the role/token is still valid. Only valid roles/tokens shall be used for access control.

5. The Hosting CSE (acting in the role of PDP) shall evaluate the access request of the Originator using access control policies and/or Role-IDs for making an access control decision as described in clause 7.1.

6. The Hosting CSE shall enforce the access control decision, i.e. either perform the resource access on behalf of the Originator or deny the resource access.

7. The Hosting CSE shall return the result of resource access back to the Originator.

7.5 Distributed Authorization

7.5.1 Introduction

The distributed authorization provides an interoperable framework between PEP, PDP, PRP and PIP when these authorization sub-components are distributed in different CSEs.

The <*authorizationDecision*>, <*authorizationPolicy*> and <*authorizationInformation*> resource types are defined to support the distributed authorization framework. The PDP, PRP and PIP procedures are bound to the <*authorizationDecision*>, <*authorizationPolicy*> and <*authorizationInformation*> resources respectively. An UPDATE operation on these resources will trigger the bound procedures. An access control decision request or access control policy request or access control information request is passed into the bound authorization procedure via the updated resource attributes. The corresponding access control decision response, access control policy response or access control information response is carried via an UPDATE response.

This clause specifies the authorization parameters exchanged between authorization sub-components and associated procedures at these authorization sub-components. The transport of distributed authorization parameters is specified in ETSI TS 118 101 [1] and ETSI TS 118 104 [4].

7.5.2 Obtain Access Control Decisions

In distributed authorization an access control decision request may be sent from one CSE to another CSE in order to obtain an access control decision from the latter. As shown in figure 7.5.2-1 there are two communication modes:

- Communication mode a: the Hosting CSE that acts as a PEP sends an access control decision request to another CSE that acts as a PDP and then receives an access control decision response from the latter.
- Communication mode b: one CSE that acts as a PDP sends an access control decision request to another CSE that acts as a PDP and then receives an access control decision response from the latter.

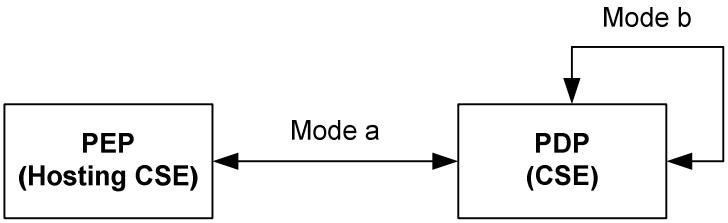


Figure 7.5.2-1: Communication modes for accessing a PDP

An access control decision requester shall send an access control decision request to a PDP via an UPDATE operation on an <*authorizationDecision*> resource. The access control decision request parameters shall be passed through updated resource attributes. The mapping between the access control decision request parameters and the corresponding resource attributes is described in table 7.5.2-1. When a valid access control decision request is passed into an <*authorizationDecision*> resource, a PDP process bound to the <*authorizationDecision*> resource shall be triggered. See clause 9.6.42 of ETSI TS 118 101 [1] for further details of <*authorizationDecision*> resource type and the PDP procedure triggering conditions. If the triggering conditions are not satisfied or there is no PDP procedure being bound to the <*authorizationDecision*> resource, the UPDATE request is treated as a normal resource by the CSE. How to bind a PDP procedure to an <*authorizationDecision*> resource is out of scope of the present document.

In the case the access control decision requester is the Hosting CSE, it obtains the address of an <*authorizationDecision*> resource from the *authorizationDecisionResourceIDs* attribute of the <*accessControlPolicy*> resource that is linked to the target resource that the Originator wants to access. See clause 9.6.2 of ETSI TS 118 101 [1] for further details of <*accessControlPolicy*> resource type. In other cases how the access control decision requester obtains the address of an <*authorizationDecision*> resource is out of scope of the present document.

Table 7.5.2-1: Mapping between the access control decision request parameters and the corresponding resource attributes of an <*authorizationDecision*>

Decision request parameter	Description	Resource attribute	Mandatory/ Optional
To	Same as the *To* parameter in table 7.1.2-1 in clause 7.1.2.	**to**	M
From	Same as the *From* parameter in table 7.1.2-1 in clause 7.1.2.	**from**	M
Operation	Same as the *Operation* parameter in table 7.1.2-1 in clause 7.1.2.	**operation**	M
Resource Type	Same as the *Resource Type* parameter in table 7.1.2-1 in clause 7.1.2.	**requestedResourceType**	O
Filter Criteria	Same as the *Filter Criteria* parameter in table 7.1.2-1 in clause 7.1.2.	**filterUsage**	O
Role IDs	Same as the *Role IDs* parameter in table 7.1.2-1 in clause 7.1.2.	**roleIDs**	O
Token IDs	Same as the *Token IDs* parameter in table 7.1.2-1 in clause 7.1.2.	**tokenIDs**	O
Tokens	Same as the *Tokens* parameter in table 7.1.2-1 in clause 7.1.2.	**tokens**	O
rq_time	Same as the *rq_time* parameter in table 7.1.2-2 in clause 7.1.2.	**requestTime**	O
rq_loc	Same as the *rq_loc* parameter in table 7.1.2-2 in clause 7.1.2.	**originatorLocation**	O
rq_ip	Same as the *rq_ip* parameter in table 7.1.2-2 in clause 7.1.2.	**originatorIP**	O

The triggered PDP process may perform the following operations:

1. Extracting the access control decision request from the updated resource attributes.
2. Retrieving applicable access control polices locally using the information provided in the access control decision request or retrieving them from another CSE using the process described in clause 7.5.3. How the PDP CSE retrieves the applicable access control policies locally or decides to contact another CSE for obtaining applicable access control policies are out of scope of the present document.
3. In the case there are Role IDs parameter and/or Token IDs parameter in the access control decision request, the PDP CSE retrieves the <role> and/or <token> resources locally or retrieves them from another CSE using the process described in clause 7.5.4. How the PDP CSE retrieves the <role> and/or <token> resources locally or decides to contact another CSE for obtaining the <role> and/or <token> resources are out of scope of the present document.
4. Evaluating the access control decision request against access control policies as described in clause 7.1.

   The PDP CSE may forward the access control decision request to another CSE that act as a PDP. How the PDP decides to do this is out of scope of the present document.
5. Updating the *decision* and *status* attributes with the access control policy evaluation result.
6. Generating an UPDATE response using the *decision* and *status* attributes and returning it back to the requester. The possible values of an access control decision returned by a PDP are listed in table 7.5.2-2. The possible values of status returned by a PDP are listed in table 7.5.2-3.
7. The PDP shall delete all the resource specific attributes after the response being sent in order to avoid information leak.

Table 7.5.2-2: Access control decision returned by a PDP

Decision value	Description
PERMIT	The PDP permits the access control decision request.
DENY	The PDP denies the access control decision request.

Table 7.5.2-3: Status returned by a PDP

Status value	Description
OK	Indicating the access control policy evaluation process is successful.
NOT_APPLICABLE	The PDP does not have any policy that applies to the access control decision request.
MISSING_ATTRIBUTE	Indicating some access control information necessary for making an access control decision is not available, e.g. roles or tokens.
SYNTAX_ERROR	Indicating there is a syntax error in access control information or an access control policy, e.g. invalid tokens or access control rules.
PROCESSING_ERROR	Indicating an error occurred during access control policy evaluation.

In the case where the *status* value is NOT_APPLICABLE, the access control decision requester should try to contact another PDP for making an access control decision if there are more than one PDP provided in the *authorizationDecisionResourceIDs* attribute of the <*accessControlPolicy*> resource, otherwise the access request of the Originator shall be denied.

7.5.3 Obtain Access Control Policies

In distributed authorization an access control policy request may be sent from one CSE to another CSE in order to obtain access control policies from the latter. As shown in figure 7.5.3-1 there are two communication modes:

- Communication mode c: one CSE that acts as a PDP sends an access control policy request to another CSE that acts as a PRP and then receives an access control policy response from the latter.
- Communication mode d: one CSE that acts as a PRP sends an access control policy request to another CSE that acts as a PRP and then receives an access control policy response from the latter.

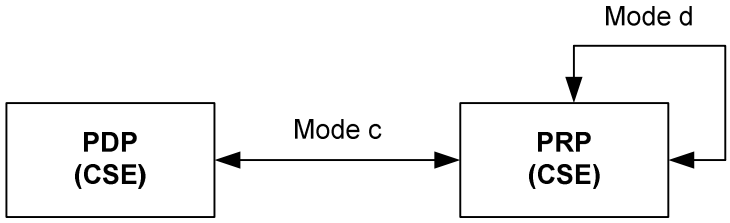


Figure 7.5.3-1: Communication modes for accessing a PRP

An access control policy requester shall send an access control policy request to a PRP via an UPDATE operation on an <*authorizationPolicy*> resource. The access control policy request parameters shall be passed through updated resource attributes. The mapping between the access control policy request parameters and the corresponding resource attributes is described in table 7.5.3-1. When a valid access control policy request is passed into an <*authorizationInformation*> resource, a PRP procedure bound to the <*authorizationPolicy*> resource shall be triggered. See clause 9.6.43 of ETSI TS 118 101 [1] for further details of <*authorizationPolicy*> resource type and the PRP procedure triggering conditions. If the triggering conditions are not satisfied or there is no PRP procedure being bound to the <*authorizationPolicy*> resource, the UPDATE request is treated as a normal resource by the CSE. How to bind a PRP procedure to an <*authorizationPolicy*> resource is out of scope of the present document.

In the case the access control policy requester is the Hosting CSE, it obtains the address of an <*authorizationPolicy*> resource from the *authorizationPolicyResourceIDs* attribute of the <*accessControlPolicy*> resource that is linked to the target resource that the Originator wants to access. See clause 9.6.2 of ETSI TS 118 101 [1] for further details of <*accessControlPolicy*> resource type. In other cases how the access control policy requester obtains the address of an <*authorizationPolicy*> resource is out of scope of the present document.

Table 7.5.3-1: Mapping between the access control policy request parameters and the corresponding resource attributes of an *<authorizationPolicy>*

Decision request parameter	Description	Resource attribute	Mandatory/ Optional
To	Same as the *To* parameter in table 7.1.2-1 in clause 7.1.2.	**to**	M

The triggered PRP process may perform the following operations:

1. Extracting the access control policy request from the updated resource attributes.
2. Retrieving applicable access control polices and policy combining algorithm locally using the information provided in the access control policy request. How the PRP CSE gets the applicable access control policies and policy combining algorithm locally is out of scope of the present document.

   The PRP CSE may forward the access control policy request to another CSE that act as a PRP, How the PRP decides to do this is out of scope of the present document.
3. Updating the *policies* and *combiningAlgorithm* attributes with the retrieval result.
4. Generating an UPDATE response using the *policies* and *combiningAlgorithm* attributes and returning it back to the requester. The possible values of a policy combing algorithm are listed in table 7.5.3-2. The possible values of an error status returned by a PRP are listed in table 7.5.3-3.
5. The PRP shall delete all the resource specific attributes after the response being sent in order to avoid information leak.

Table 7.5.3-2: Policy combining algorithm returned by a PRP

Decision	Description
PERMIT_OVERRIDES	If an access request is permitted by any access control policy, then the access request is permitted.

Table 7.5.3-3: Status returned by a PRP

Status value	Description
OK	Indicating the access control policy retrieval process is successful.
NOT_APPLICABLE	The PRP does not have any policy that applies to the access control policy request.
PROCESSING_ERROR	Indicating an error occurred during retrieving access control policy.

In the case where the *status* value is NOT_APPLICABLE, the access control policy requester should try to contact another PRP for retrieving access control policies if there are more than one PRP provided in the *authorizationPolicyResourceIDs* attribute of the <*accessControlPolicy*> resource, otherwise the access request of the Originator shall be denied.

7.5.4 Obtain Access Control Information

In distributed authorization an access control information request may be sent from one CSE to another CSE in order to obtain access control information from the latter. As shown in figure 7.5.4-1 there are two communication modes:

- Communication mode e: one CSE that acts as a PDP sends an access control information request to another CSE that acts as a PIP and then receives an access control information response from the latter.
- Communication mode f: one CSE that acts as a PIP sends an access control information request to another CSE that acts as a PIP and then receives an access control information response from the latter.

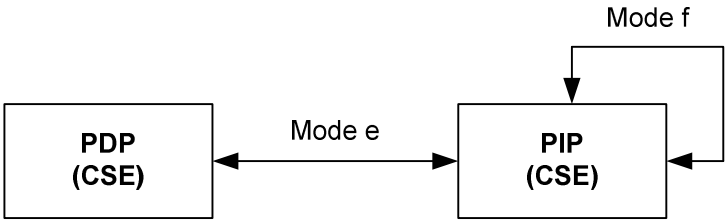


Figure 7.5.4-1: Communication modes for accessing a PIP

An access control information requester shall send an access control information request to a PIP via an UPDATE operation on an <*authorizationInformation*> resource. The access control information request parameters shall be passed through updated resource attributes. The mapping between the access control information request parameters and the corresponding resource attributes is described in table 7.5.4-1. When a valid access control policy request is passed into an <*authorizationInformation*> resource, a PIP procedure bound to the <*authorizationInformation*> resource will be triggered. See clause 9.6.44 of ETSI TS 118 101 [1] for further details of <*authorizationInformation*> resource type and the PIP procedure triggering conditions. If the triggering conditions are not satisfied or there is no PIP procedure being bound to the <*authorizationInformation*> resource, the UPDATE request is treated as a normal resource by the CSE. How to bind a PIP procedure to an <*authorizationInformation*> resource is out of scope of the present document.

In the case the access control information requester is the Hosting CSE, it obtains the address of an <*authorizationInformation*> resource from the *authorizationInformationResourceIDs* attribute of the <*accessControlPolicy*> resource that is linked to the target resource that the Originator wants to access. See clause 9.6.2 of ETSI TS 118 101 [1] for further details of <*accessControlPolicy*> resource type. In other cases how the access control information requester obtains the address of an <*authorizationInformation*> resource is out of scope of the present document.

Table 7.5.4-1: Mapping between the access control information request parameters and the corresponding resource attributes of an *<authorizationInformation>*

Decision request parameter	Description	Resource attribute	Mandatory/ Optional
From	Same as the *From* parameter in table 7.1.2-1 in clause 7.1.2.	**from**	M
Role IDs	Same as the *Role IDs* parameter in table 7.1.2-1 in clause 7.1.2.	**roleIDs**	O
Token IDs	Same as the *Token IDs* parameter in table 7.1.2-1 in clause 7.1.2.	**tokenIDs**	O

The triggered PIP process may perform the following operations:

1. Extracting the access control information request from the updated resource attributes.
2. Retrieving applicable access control information locally using the information provided in the access control information request. How the PIP CSE gets the requested <*role*> and/or <*token*> resources locally is out of scope of the present document.

   The PIP CSE may forward the access control information request to another CSE that act as a PIP, How the PIP decides to do this is out of scope of the present document.
3. Updating the <*role*> and <*token*> resources with the retrieval result.
4. Generating an UPDATE response using the <*role*> and/or <*token*> child resources and returning it back to the requester. The possible values of an error status returned by a PIP are listed in table 7.5.4-2.
5. The PIP shall delete all the resource specific attributes and child resources after the response being sent in order to avoid information leak.

Table 7.5.4-2: Status values returned by a PIP

Status value	Description
OK	Indicating the access control information retrieval process is successful.
NOT_APPLICABLE	The PIP does not have any role or token that applies to the access control information request.
PROCESSING_ERROR	Indicating an error occurred during retrieving access control information.

In the case where the *status* value is NOT_APPLICABLE, the access control information requester should try to contact another PIP for retrieving access control information if there are more than one PIP provided in the *authorizationInformationResourceIDs* attribute of the <*accessControlPolicy*> resource, otherwise the access request of the Originator shall be denied.

7.5.5 Distributed Authorization Resource Lifecycle

<*authorizationDecision*>, <*authorizationPolicy*> and <*authorizationInformation*> are all the child resources of a <*CSEBase*> resource. As these resources are related to authorization functions, their creation and maintenance shall be tightly controlled by the authorized entities (e.g. a security administrator). The authorization resource management permissions (e.g. resource creation, update and deleting) shall be specified by the access control policies assigned to the <*CSEBase*> resource. The access control policies which specify who can send authorization requests to the authorization resources shall be assigned to the authorization resources directly.

Before an authorization resource is bound to an authorization process (i.e. a PDP, PRP or PIP process), it acts as a normal resource to a Create (C), Retrieve (R), Update (U), Delete (D) or Notify (N) operation. After being bound to an authorization process, an UPDATE operation on this resource may trigger the bound process. How to bind an authorization process to these authorization resources is out of scope of the present document.

An entity that needs to send authorization requests to an authorization resource shall only have the update permission on the authorization resource specific attributes and/or child resources.

8 Security Frameworks

8.1 General Introductions to the Security Frameworks

8.1.0 General

To accommodate the variety of deployment scenarios that can be encountered in M2M applications, the present document supports a diversity of methods to provision and establish security in M2M systems.

8.1.1 General Introduction to the Symmetric Key Security Frameworks

In the Symmetric Key Security Frameworks, each pair of entities that need to authenticate each other is provisioned with its own shared symmetric key. This is performed through pre-provisioning, e.g. during device manufacturing or deployment, or a remote security provisioning framework.

8.1.2 General Introduction to the Certificate-Based Security Frameworks

8.1.2.0 Introduction

This clause describes the Credential Configuration and Certificate Verification used in the Certificate-Based Security Association Establishment Framework and Certificate-Based Remote Security Provisioning Framework.

8.1.2.1 Public Key Certificate Flavours

The present document defines procedures using the following Public Key Certificate flavours:

- Raw Public Key Certificates:
  - **Description:** A raw public key certificate (IETF RFC 7250 [37]) contains only the raw public key, without other information normally provided in a certificate. The raw public key certificate is exchanged in the TLS handshake in the place of a traditional certificate (see IETF RFC 7250 [37]).
  - **Use:** A raw public key certificate can be used for authenticating a CSE or AE either during the Association Security Handshake phase of the Certificate-Based Security Association Establishment or during the Bootstrap Enrolment Handshake phase of the Certificate-Based Remote Security Provisioning Framework.
- Device certificates:
  - **Description:** These certificates have a certificate chain to a trust anchor and include one or more globally unique hardware instance identifier (such as the Object Identifier Based M2M Device identifiers discussed in annex H "Object Identifier Based M2M Device Identifier" ETSI TS 118 101 [1]) in the subjectAltName extension of the certificate. A device certificate can be used to verify the identity of the hardware instance on which the entity is being executed.
  - **Use:** Device certificates can be used to authenticate a CSE or AE executing on a specific M2M Device. If the M2M device is an ASN or MN (which supports a CSE), then the device certificate is implicitly associated with the CSE that executes on the device. If the device is an ADN (which does not support a CSE) then the device certificate is not implicitly associated with a specific AE executing on the hardware. A device certificate can be used for authenticating a Field Domain CSE either during the Association Security Handshake phase in the Certificate-Based Security Association Establishment Framework or during the Bootstrap phase of the Certificate-Based Remote Security Provisioning Framework.
- Node-ID certificates:
  - **Description:** These certificates have a certificate chain to a trust anchor and include the Node-ID of a Node (see ETSI TS 118 101 [1]) in the subjectAltName extension of the certificate. A Node-ID certificate can be used to verify the identity of a Node.
  - **Use:** A Node-ID certificate can be used to authenticate a Security Principal on a Node acting on behalf of the CSE and/or AE(s) executing on a specific Node. If the Node supports a CSE i.e. an ASN or MN, then the Node-ID certificate is implicitly associated with the CSE that executes on the Node. If the Node does not support a CSE i.e. an ADN, then the Node-ID certificate is not implicitly associated with a specific AE executing on the Node.
- CSE-ID certificates:
  - **Description:** These certificates have a certificate chain to a trust anchor and include the public domain name representation of a CSE-ID (see ETSI TS 118 101 [1]) in the subjectAltName extension of the certificate. A CSE-ID certificate verifies that the entity presenting the certificate has been assigned a particular CSE-ID.
  - **Use:** A CSE-ID certificate can be used to authenticate a CSE only.
- AE-ID certificates:
  - **Description:** These certificates have a certificate chain to a trust anchor and include the full URI representation of an AE-ID in the subjectAltName extension of the certificate. An AE-ID certificate verifies that the entity presenting the certificate has been assigned a particular AE-ID.
  - **Use:** An AE-ID certificate can be used to authenticate an AE only.

- FQDN certificates:

  - **Description:** These certificates have a certificate chain to a trust anchor and include the FQDN of an M2M Enrolment Function in the subjectAltName extension of the certificate. An FQDN certificate verifies that the entity presenting the certificate has been assigned a particular FQDN.

  - **Use:** A FQDN certificate is used to authenticate an M2M Enrolment Function to an Enrolee during a Bootstrap Enrolment Handshake phase in a Certificate-Based Remote Security Provisioning Framework.

NOTE: The flavours, and the details specific for these flavours, are specified to support a range of deployment models while ensuring that oneM2M entities have clear procedures for authenticating other oneM2M entities using certificates.

The profiles for these certificates are found in clause 10.1.1.

8.1.2.2 Certification Path Validation and Certificate Status Verification

If an entity is to authenticate another entity using a device certificate, CSE-ID certificate, AE-ID certificate, Node-ID certificate or FQDN certificate, then the entity shall perform basic certification path validation (section 6.1of IETF RFC 5280 [34]) as part of verifying the other entity's certificate (see clause 8.1.2.4).

CA certificates shall include the name constraint extensions (section 4.2.1.10 of IETF RFC 5280 [34]) and shall constrain the names (object identifier M2M Device IDs from annex H "Object Identifier Based M2M Device Identifier" of ETSI TS 118 101 [1], public domain name representation of the CSE-ID, Absolute AE-ID, Node-ID or FQDNs) which may be in the subsequent certificate used to authenticate the entity (device certificate, CSE-ID certificate, AE-ID certificate, Node-ID certificate or FQDN certificate respectively).

- Section 4.2.1.10 in IETF RFC 5280 [34] describes how the name constraint extension is used for constraining URIs and FQDNs.

- Section 10.4.1.4.2 in IETF RFC 5280 [34] describes how the name constraint extension is used for constraining object identifier M2M Device IDs.

The trust anchor information (section 6.1.1 of IETF RFC 5280 [34]) is provided to the entity during Credential Configuration, Association Configuration, Bootstrap Credential Configuration or Bootstrap Instruction Configuration.

NOTE 1: Section 6.1.1 of IETF RFC 5280 [34] states "The trust anchor information is trusted because it was delivered to the path processing procedure by some trustworthy out-of-band procedure". Credential Configuration, Association Configuration, Bootstrap Credential Configuration and Bootstrap Instruction Configuration satisfy the requirements of being trustworthy out-of-band procedures.

Certificate status verification: In the case of an Infrastructure Domain entity receiving an MEF certificate, the entity shall verify the status of the certificate using a Certificate Revocation List as described in IETF RFC 5280 [34]. A mapping of the Online Certificate Status Protocol (OCSP) onto HTTP may be used, as described in Appendix A of IETF RFC 6960 [35], however a mapping of OCSP onto CoAP is not currently defined. Furthermore, OCSP may also not be easily applicable in all environments. An alternative approach may be using the TLS Certificate Status Request extension (section 8 of IETF RFC 6066 [44]; also known as "OCSP stapling") or preferably the Multiple Certificate Status Extension (IETF RFC 6961 [36]), if available.

NOTE 2: Most of the above paragraph is based on almost identical text in the CoAP specification IETF RFC 7252 [i.21], a protocol with similar (if not identical) considerations to oneM2M deployments.

8.1.2.3 Credential Configuration for Certificate-Based Security Framework

If an entity is to authenticate itself using a Certificate-Based Security Framework, then the entity shall be pre-provisioned with the following information:

- The entity's Private Signing Key, which should remain protected in a secure environment, e.g. using the framework described in annex L.

NOTE: An entity authenticates itself to other entities by proving that it knows the Private Signing Key corresponding to a particular Public Verification Key.

- The entity's Certificate (and if applicable, Certificate Chain) as described in clause 10.1.1.

- In the case of a CSE-ID certificate the entity shall be configured with the entity's CSE-ID.
- In the case of an AE-ID certificate the entity shall be configured with the entity's AE-ID.

8.1.2.4 Information Needed for Certificate Authentication of another Entity

Entity A shall be configured to trust the following information in order to authenticate Entity B using the certificate-Based SAEF:

- An indication of the public key certificate flavour of other Entity B's Certificate (that is, raw public key certificate, device certificate, CSE-ID certificate, Node-ID certificate or FQDN certificate).
- In the case where Entity B's certificate is a raw public key certificate:
  - A public key identifier for the raw public key in the certificate (see clause 10.1.2).
- In the case where other Entity B's certificate is a device certificate, CSE-ID certificate, Node-ID certificate or FQDN certificate:
  - **A Globally unique identifier:** The globally unique identifier for the entity which is also present in the subjectAltName extension of the other entity's certificate:
    - Device Certificate: A globally unique hardware instance identifier (such as the object identifier M2M Device ID in annex H "Object Identifier Based M2M Device Identifier" ETSI TS 118 101 [1]) that is present in the device certificate.
    - CSE-ID Certificate: The public domain name representation of the CSE-ID as defined in ETSI TS 118 101 [1].
    - Node-ID Certificate: The Node-ID of a Node as defined in ETSI TS 118 101 [1].
  - **Trust Anchor Information:** For the trust anchor certificates of Entity B's certificate chain (see clause 8.1.2.2).

Entity B shall be configured to trust the following information in order to authenticate Entity A using the Certificate-Based SAEF:

- An indication of the public key certificate flavour of Entity A's Certificate (that is, raw public key certificate, device certificate, CSE-ID certificate, Node-ID certificate or AE-ID Certificate).
- In the case where Entity A's certificate is a raw public key certificate:
  - A public key identifier for the raw public key in the certificate (see clause 10.1.2).
- In the case where Entity A's certificate is a device certificate, CSE-ID certificate, Node-ID certificate or AE-ID certificate:
  - **Trust Anchor Information:** for the trust anchor certificate for Entity A's certificate chain (see clause 8.1.2.2).

In order to authenticate the M2M Enrolment Function using the certificate-based RSPF, an Enrolee shall be configured to trust the trust anchor information of the M2M Enrolment Function's certificate chain.

An M2M Enrolment Function shall be configured to trust the following information in order to authenticate an Enrolee using the certificate-based RSPF:

- An indication of the public key certificate flavour of Entity B's Certificate (that is, raw public key certificate or device certificate).
- In the case where the Enrolee's certificate is a raw public key certificate:
  - A public key identifier for the raw public key in the certificate (see clause 10.1.2).

- In the case where the Enrolee's certificate is a device certificate, CSE-ID certificate, Node-ID certificate or AE-ID certificate:
  - **A Globally unique identifier:** The globally unique identifier which is also present in the subjectAltName extension of the Enrolee's certificate:
    - Device Certificate: A globally unique hardware instance identifier (such as the object identifier M2M Device ID in annex H "Object Identifier Based M2M Device Identifier" ETSI TS 118 101 [1]) that is present in the device certificate.
    - CSE-ID Certificate: The public domain name representation of the CSE-ID as defined in ETSI TS 118 101 [1].
    - Node-ID Certificate: The Node-ID of a Node as defined in ETSI TS 118 101 [1].
    - AE-ID Certificate: The Absolute AE-ID assigned to the AE.
  - **Trust Anchor Information:** for the trust anchor certification for the Enrolee's certificate chain (see clause 8.1.2.2).

8.1.2.5 Certificate Verification

This clause describes how an entity authenticates the other entity in the Security Handshake of a Certificate-Based Security Framework.

The other entity's Certificate is received during the Security Handshake.

The other entity's Certificate is verified as follows:

- If the certificate information configured during the Association Configuration or Bootstrap Instruction Configuration indicates that the other entity's Certificate is a raw public key certificate, then the entity verifies that the public key identifier (received during Association Configuration or Bootstrap Instruction Configuration) corresponds to the raw public key certificate (received during the Security Handshake) using the process described in clause 10.1.2.
- If the certificate information configured during the Association Configuration or Bootstrap Instruction Configuration indicates that the other entity's Certificate is a device certificate, CSE-ID certificate, AE-ID certificate, Node-ID certificate or FQDN certificate, then the entity shall perform the following verifications:
  - The entity shall look for a match between the globally unique identifier described in clause 8.1.2.4 (received during Association Configuration or Bootstrap Instruction Configuration) and the values in the subjectAltName extension of the other entity's Certificate (received during the Security Handshake). If there is not an exact match, then the entity shall abort the (D)TLS handshake:
    - In the case of device certificate, the globally unique identifier is a globally unique hardware instance identifier (such as the object identifier M2M Device ID in annex H "Object Identifier Based M2M Device Identifier" ETSI TS 118 101 [1]). In this case, the notion of a "match" depends on how the globally unique hardware instance identifier may be represented in the subjectAltName extension.
    - In the case of a CSE-ID certificate, the globally unique identifier is the public domain name representation of the CSE-ID as defined in ETSI TS 118 101 [1], and a match is a FQDN in the subjectAltName extension in the other entity's certificate that is an exact match for the public domain name representation of the CSE-ID.
    - In the case of an AE-ID certificate, the globally unique identifier is the AE-ID, and a match is a URI in the subjectAltName extension in the other entity's certificate that is an exact match for the Absolute AE-ID.
    - In the case of a Node-ID certificate, the globally unique identifier is the M2M-Node-ID as defined in ETSI TS 118 101 [1], and a match is a Node-ID in the subjectAltName extension in the other entity's certificate that is an exact match for the Node-ID.

- In the case of an FQDN certificate, the globally unique identifier is the FDQN of the M2M Authentication Function or M2M Enrolment Function, and a match is a URI, FQDN or dNSName in the subjectAltName extension in the other entity's certificate that is an exact match for the FDQN of the M2M Authentication Function or M2M Enrolment Function.

- The entity shall perform path validation and certificate status verification using the trust anchor certificate as described in clause 8.1.2.2). If this verification fails, then the entity shall abort the (D)TLS handshake.

NOTE: After a successful Security Handshake in which the other entity provides a Certificate Chain, the other entity's identity (received during Association Configuration or Bootstrap Instruction Configuration) can be associated with additional information extracted from the other entity's Certificate Chain (e.g. the other entity manufacturer, other entity owner, or conformance criteria). These details are not described in the present document.

8.1.3 General Introduction to the GBA (Generic Bootstrapping Architecture) Framework

Generic Bootstrapping Architecture (GBA) is a framework that could be used for Remote Security Provisioning.

In case of scenario where the M2M Service Provider and the operator of the underlying network have an agreement to use the underlying network credentials as the basis for security between a M2M Application Service/Middle Node and Infrastructure Node (including the case that the M2M Service Provider and the operator of an underlying network are actually the same entity), GBA procedure could be used.

It is important that this feature is used only within the scope of an appropriate agreement between the M2M Service Provider and the operator of the underlying network. The normative text for the GBA-Based Security Association Establishment Framework (clause 8.2.2.2) and the GBA-Based Security Bootstrap Framework (clause 8.3.2.2) implicitly assumes that such an agreement is already in place. Since the present document is a technical specification, it does not address the details of such an agreement.

A general introduction to GBA is included in ETSI TR 118 508 [i.4].

After a successful GBA bootstrapping, the M2M Application Service/Middle Node and the BSF share a security association which consists of a bootstrapping transaction identifier (B-TID) and key material (GBA bootstrap Ks).

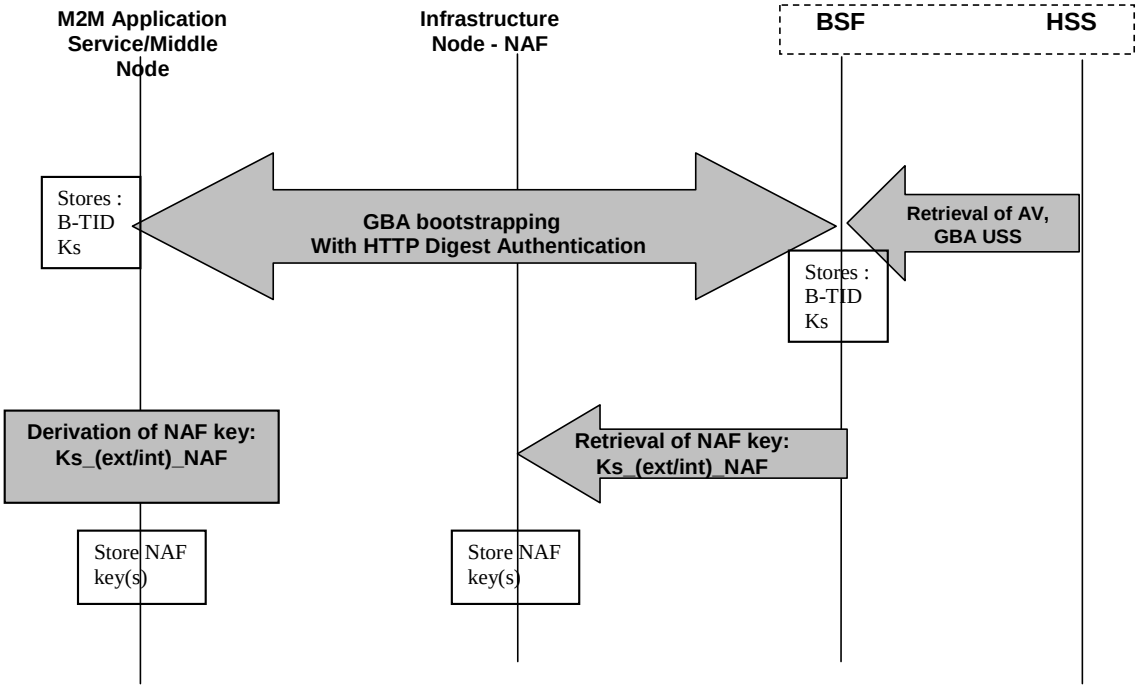
This security association can be used by the M2M Application Service/Middle Node to derive NAF keys (Ks_(ext/int)_NAF) shared between a M2M Application Service/Middle Node and a M2M Infrastructure Node or an M2M Authentication Function.

There are two modes of GBA: ME-based GBA (GBA_ME) and UICC-based GBA (GBA_U). In case of GBA_ME, one NAF-specific key is derived: the key Ks_NAF. In case of GBA_U, two NAF-specific keys are derived: Ks_ext_NAF (available in the ME) and Ks_int_NAF (which remains inside the UICC).

GBA_U can be performed only when the UICC is GBA aware.

The BSF determines which mode to run based on the UICC capability indicated in the GBA User Security Settings (GUSS).

The usage of GBA_U is recommended since it provides a higher level of security than GBA_ME. The implication of this recommendation is that the entity, AE or CSE, using the GBA_U-based NAF keys should be resident in the UICC.



NOTE: Network Application Function (NAF) may be an Infrastructure Node or an M2M Authentication Function.

Figure 8.1.3-1: GBA framework

8.2 Security Association Establishment Frameworks

8.2.1 Overview on Security Association Establishment Frameworks

The Security Association Establishment Frameworks (SAEF) described in the present document, apply to direct connections on the Mcc, Mcc' or Mca reference points.

The Security Association Establishment Framework end-points are denoted:

- Entity A, which may be an AE or CSE. This entity always acts as the client of the security association (TLS/DTLS session).
- Entity B, which shall be a CSE. This entity always acts as the server of the security association (TLS/DTLS session).

If entity A is request reachable, i.e. capable to receive request primitives from entity B on a separate point of access (see clauses 9.3.2 and 9.6.5 of ETSI TS 118 101 [1]), it also acts as the server of a second security association (TLS/DTLS session denoted SA2) and entity B acts as a client of this second security association. The relationship between the first and second security association, SA1 and SA2, is specified below and illustrated in figure 8.2.1-2.

The oneM2M system supports the following Security Association Establishment Frameworks:

- **Security Association Establishment Frameworks:**
  - **Provisioned Symmetric Key** Security Association Establishment. A symmetric key is provisioned to the entities: this is called the Provisioned M2M Symmetric Key, and denoted Kpsa. The entities authenticate each other by verifying Message Integrity Codes (MIC) in the Security Handshake which were generated using the symmetric key. For more details see clause 8.2.2.1.
  - **Certificate-Based** Security Association Establishment: The entities are each issued with:
    - a Private Signing Key that is known only to that entity;
    - a Certificate containing the corresponding Public Verification Key; and

- (Optionally) a Certificate Chain from the entity's Certificate to a Root Certificate.

  The entities validate each other's Certificate before trusting the Public Verification Keys in the Certificate. Within the Security Handshake, entity A creates a digital signature of the session parameters using its private signing key and entity B verifies the digital signature using entity A's public verification key. Then the roles are reversed: entity B creates a digital signature and entity A verifies it. For more details, see clause 8.2.2.2.

- **M2M Authentication Function (MAF)-based Security Association Establishment.** This Security Association Establishment Framework uses mutual authentication of the entity A and the M2M Authentication Function (MAF) and derive a M2M Secure Connection key (Kc) that the MAF delivers to entity B (via separate mutually-authenticated communication). The entities then authenticate each other using the M2M Secure Connection key (Kc). Each of Entity A and Entity B can use either symmetric key credentials or certificates for mutual authentication with the MAF. For more details see clause 8.2.2.3.

For a more detailed description of the above Security Association Establishment Frameworks, it is useful to compare the following aspects of the Security Association Establishment Frameworks:

- **Credential Configuration:**

  - For the Provisioned Symmetric Key Security Association Establishment Framework, Entity A and Entity B are provisioned with the Provisioned M2M Symmetric Key that entities subsequently use to authenticate each other using pre-provisioning or remote provisioning.

  - For the Certificate-Based Security Association Establishment Frameworks, Entity A and Entity B are pre-provisioned with the Credential that the entity subsequently use to authenticate itself to the other entity using pre-provisioning or remote provisioning.

  - For the MAF-based Security Association Establishment Framework, the MAF Credential Configuration procedure (clause 8.8.3.1) is performed twice: once to provision credentials for mutual authentication of Entity A with MAF, and once to provision credentials for mutual authentication of Entity B with MAF.

  The method for pre-provisioning of credentials for mutual authentication can be deployment dependent. Interoperable frameworks enabling pre-provisioning are described in annex D for UICC and in annex L for independent hardware based secure environments supporting asymmetric cryptography. Mechanisms for remote provisioning are specified in clause 8.3. This includes remote provisioning of credentials using the device configuration mechanisms specified in ETSI TS 118 122 [57].

- **Identity Configuration:** Identity configuration can occur as part of Credential Configuration, or can occur at a later time.

- For the MAF-based Security Association Establishment Framework, the MAF is configured with information about the identities of Entity B and, optionally, Entity A. Clause 8.2.2.3 provides additional details.

NOTE 1: The current oneM2M specifications do not describe how this information is configured to the MAF.

  - Entity A's knowledge of its identity (IdA) has no impact on the security association establishment.

  - Entity B shall be configured with its CSE-ID (IdB) prior to Association Configuration.

- **Association Configuration:**

  - Entity A shall be provided with IdB, the CSE-ID for Entity B.

NOTE 2: The present document does not describe how Entity A is provided with IdB. Example mechanisms could include configuration via remote management, and discovery mechanisms supported by the Underlying Network(s).

  - In the case of Certificate-Based Authentication Framework: each entity (Entity A and Entity B) is additionally configured with the certificate information that the entity subsequently uses to verify the other entity. The necessary certificate information is dependent on the flavour of the certificates, with details provided in clause 8.1.2.4.

- In the case of the MAF-Based Security Association Establishment Framework:

    1. The MAF is provided with the identity of Entity B for which the MAF is authorized to facilitate establishing a security association with Entity A.

    2. Entity A and the MAF interact, using the MAF Key Registration procedure (clause 8.8.2.7) to establish M2M Secure Connection Key (Kc) and M2M Secure Connection Key Identifier (KcID) and authorize Entity A to establish a security association with Entity B. This step includes mutual authentication using the MAF Handshake Procedure (clause 8.8.2.2). This step includes Entity A providing the MAF with IdB. See note 2 above,

- **Association Security Handshake:** Identification, authentication and security context establishment between the entities:

    - In the case of the MAF-based Security Association Establishment Framework:

        1. Entity A provides the M2M Secure Connection Key Identifier (KcID) to Entity B.

        2. Entity B and the MAF interact using the MAF Key Retrieval procedure (clause 8.8.2.8). This step includes mutual authentication using the MAF Handshake Procedure (clause 8.8.2.2). Entity B forwards KcID to the MAF and, if Entity B is authorized, the MAF returns the M2M Secure Connection Key (Kc) and either IdA or a globally unique identifier for the credential used by the MAF to authenticate Entity A during Association Configuration.

        3. The M2M Secure Connection Key (Kc) is then used in the Security Handshake for mutual authentication between Entity A and Entity B.

    Entity A associates the resulting security context with IdB: the AE-ID or CSE-ID for Entity B established during Association Configuration.

    Entity B associates the security context with one of the following:

    - A single Absolute CSE-ID, and indication that Entity A is a CSE;

    - A single Absolute AE-ID, and indication that Entity A is an AE; or

    - A list of allowed Absolute AE-ID values, and indication that Entity A is an AE. This case applies only when Entity A presents a Device Certificate or a Node-ID Certificate.

    The present document provides the following approaches for Entity B to determine the applicable CSE-ID or AE-ID(s) prior to registration:

    - If Entity A is authenticated using a CSE-ID certificate (or AE-ID certificate), then Entity B extracts the CSE-ID (or AE-ID respectively) from the certificate and associates the security context with this CSE-ID (or AE-ID respectively), as described in the certificate profile in clause 10.1.1.

    - In all other cases, Entity B forms a globally unique Credential-ID (see clause 10.4) identifying the credential used by Entity A in the security association establishment mechanism. The Credential-ID identifies one of a Kpsa (in the case of a PSK SAEF), certificate (in the case of a Certificate-Based SAEF) or the Km (in the case of an MAF-Based SAEF). Entity B subsequently determines the CSE-ID or AE-ID(s) which are applicable for this Credential-ID.

        - If Entity B assigned the AE-ID(s) corresponding to this Credential-ID, then Entity B is responsible for determining the AE-ID(s) corresponding to this Credential-ID.

        - Otherwise, the CSE-ID or AE-ID(s) can be made available to Entity B via one of the following approaches. The M2M SP is expected to ensure one of these approaches will successfully provide the CSE-ID or AE-ID(s) of Entity A.

- If the Security Association Establishment procedure is facilitated by an M2M Authentication Function, then the M2M Authentication Function may be provided with the CSE-ID or AE-ID and the M2M Authentication Function may provide this to Entity B at the same time as Kc is provided to Entity B. The M2M Authentication Function could have been provided with the CSE-ID or AE-ID during provisioning, including the case where the M2M Authentication Function is provided with the CSE-ID or AE-ID during remote provisioning by an M2M Enrolment Function (which is similar to the case described in the following bullet).

- If the Security Association Establishment procedure uses a Provisioned Symmetric Key which was remotely provisioned to Entity A and Entity B, then the M2M Enrolment Function may provide Entity B with CSE-ID or AE-ID during the Remote Security Provisioning procedure.

- If the M2M Service Provider assigns Entity A's entity identifier(s), then the CSE-ID or AE-ID(s) may be securely configured by the M2M Service Provider to Entity B prior to the Association Security Handshake. For example, the CSE-ID or AE-ID(s) may be configured as part of Credential Configuration or Association Configuration. The present document permits using other mechanisms, with the assumption that the mechanism provides authentication, integrity protection and optionally confidentiality.

EXAMPLE 1: If the M2M Service Provider has the opportunity to configure Entity B prior to deployment, then the M2M Service Provider could configure the CSE-ID or AE-ID(s) to Entity B at this time.

EXAMPLE 2: A secure remote management protocol could be used to configure Entity B with the CSE-ID or AE-ID(s). However, this is not currently an interoperable feature as there is no standardized management object facilitating this management.

- In the case that Entity A is an AE and Entity B is a CSE, the applicable AE-ID(s) may be obtained by retrieving the applicable <*serviceSubscribedAppRule*> resources which are linked to by the *ruleLinks* attribute of the Entity B's <*serviceSubscribedNode*> on the IN-CSE as described in clause 10.1.1.2.2 in ETSI TS 118 101 [1].

Figure 8.2.1-1 provides a summary of the above defined three Security Association Establishment Frameworks.

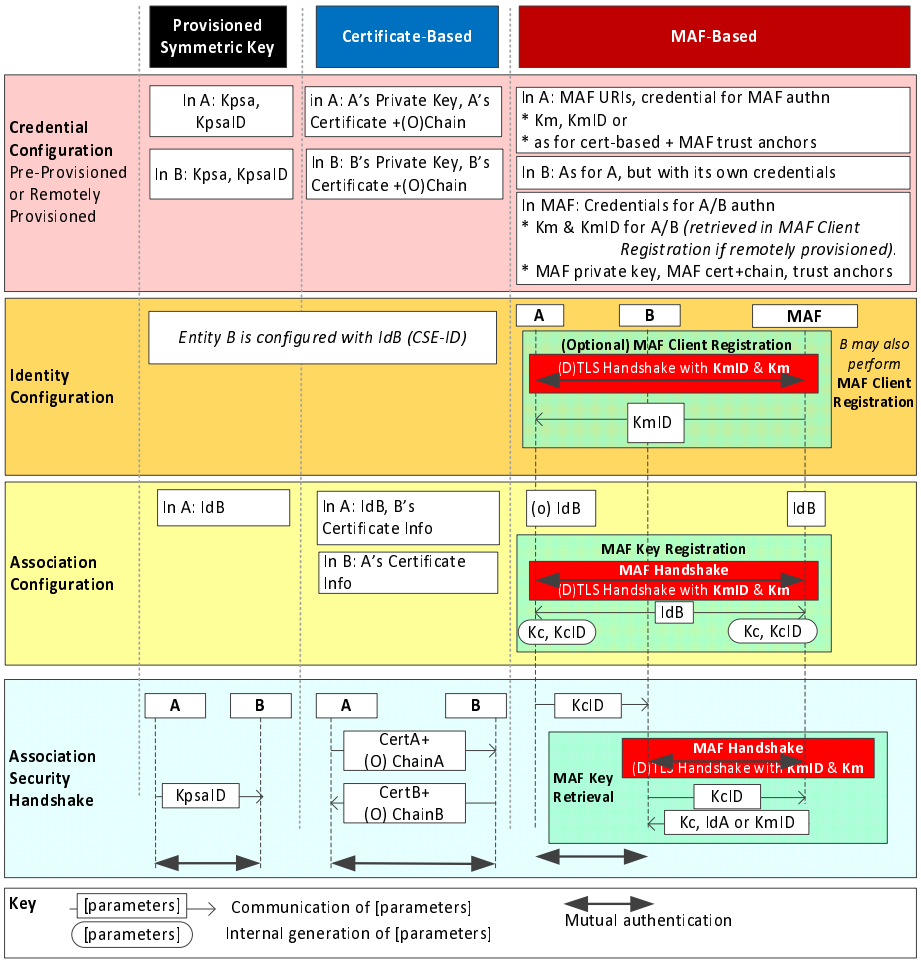


Figure 8.2.1-1: Overview of the Security Association Establishment Frameworks supported by oneM2M

If entity A is request reachable, i.e. capable to receive request primitives from entity B on a separate point of access (see clauses 9.3.2 and 9.6.5 of ETSI TS 118 101 [1]), then a second security association needs to be established between entity A and entity B to serve secure communications in the reverse direction.

In order to allow differentiation between the two security associations between a pair of entities, SA1 and SA2 are used to denote them in the following. SA1 refers to the security association established when entity A acts as the registree which sends requests to its registrar CSE. SA2 refers to the security association established when the registrar entity B sends requests to its request reachable registree entity A.

Figure 8.2.1-2 depicts the sequence of steps when establishing SA1 and SA2. Since the request reachable entity A can receive requests only after it has registered to entity B, security association SA1 has to be established always prior to first-time establishment of SA2. Establishment of security association SA1 is performed as outlined in figure 8.2.1-1, using one of the applicable Security Association Establishment Frameworks, i.e. provisioned symmetric key (PSK) based, certificate based or MAF based SAEF. The four phases described above and illustrated in figure 8.2.1-2 are executed: credential configuration, identity configuration, association configuration and association security handshake. The details of these procedures are specified in clause 8.2.2.

When security association SA2 is established subsequently, the procedure can take advantage of already available credential configuration, identity configuration and association configuration and does not need to execute these steps again. SA2 establishment reduces to the association security handshake step, i.e. to performing a (D)TLS handshake using the credentials established with security association SA1. When MAF-based SAEF is applied, there is no need to execute the MAF Key retrieval procedure unless the credentials are expired. The symmetric key credentials Kc and KcID established with SA1 can be used directly to perform the (D)TLS PSK handshake.

Note that for establishment of security association SA2 the roles of entities A and B are reversed compared to SA1 establishment. In SA2, entity B acts as (D)TLS client and entity A acts as (D)TLS server. There is otherwise no difference.

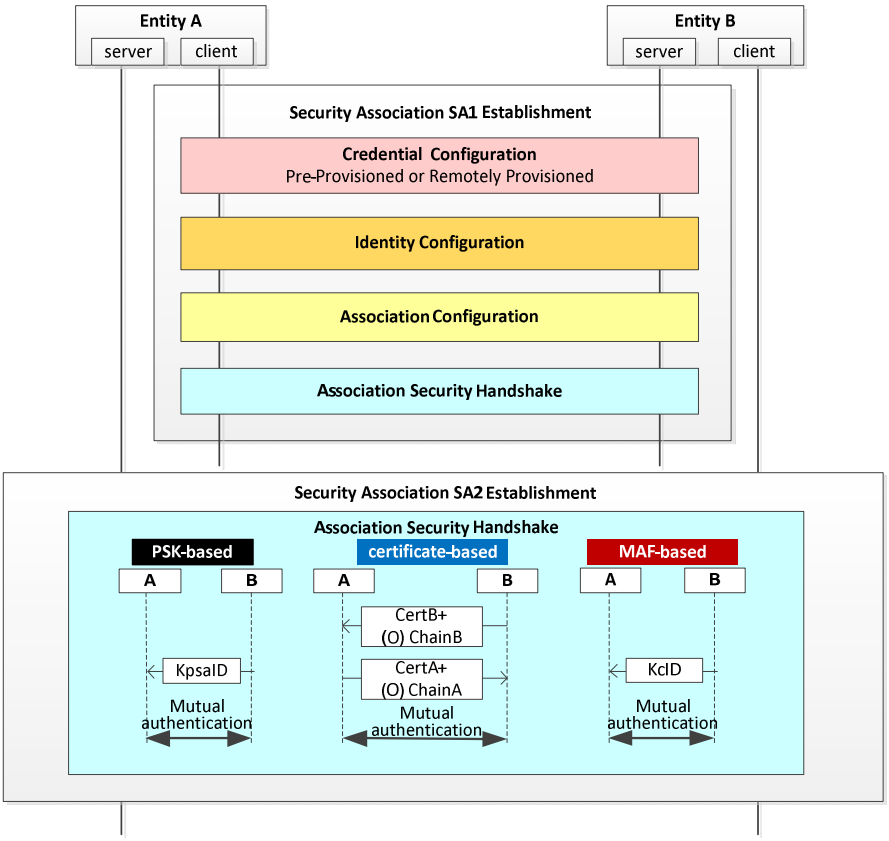


Figure 8.2.1-2: Security Associations for request reachable entities

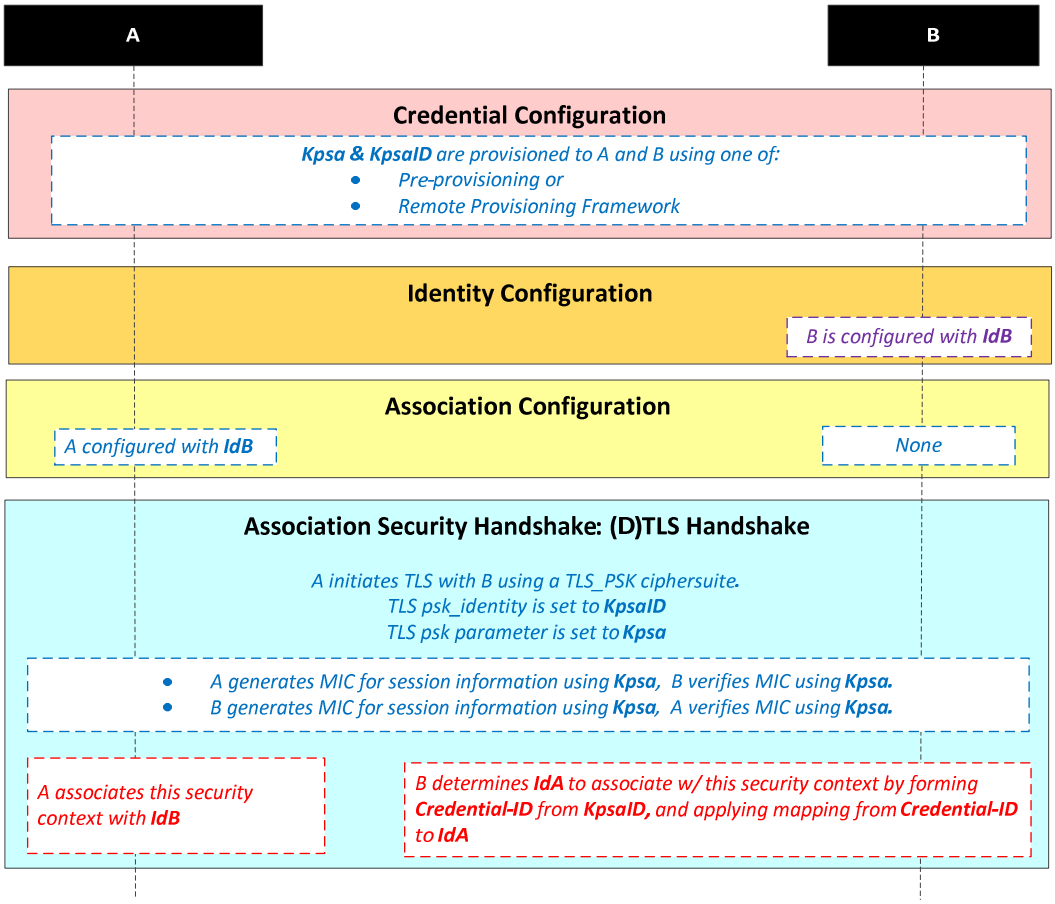
8.2.2 Detailed Security Association Establishment Frameworks

8.2.2.1 Provisioned Symmetric Key Security Association Establishment Frameworks

This clause describes the Provisioned Symmetric Key Security Association Establishment Framework. This framework enables mutual authentication of two entities corresponding to either two CSEs or a CSE and an AE. The Credential for this framework is a long-term symmetric key that has been provisioned into the entities to be authenticated. This key is called a Provisioned Secure Connection Key and is denoted Kpsa. The provisioning of Kpsa could be a pre-provisioning or a remote provisioning thanks to Remote Security Provisioning Frameworks, as described in clause 8.3. The entities authenticate each other by verifying message authentication codes in the Association Security Handshake which were generated using the Provisioned Secure Connection Key.

NOTE: Long term Provisioned Secure Connection Keys can pose a security risk if not adequately secured, and for this reason Long Term Provisioned Secure Connection Keys are recommended to be stored in Secure Environments.

Figure 8.2.2.1-1 illustrates the sequence of events when using the Provisioned Symmetric Key Security Association Establishment Framework. In this description, "Entity A" and "Entity B" correspond to either two CSEs or a CSE and an AE or an AE and a CSE (respectively).



NOTE: The following font colours differentiate the general topic that the text relates to:
Blue italic text highlights details specific to this particular Security Association Establishment Framework.
Purple italic text highlights technical actions that may include steps not specified by oneM2M.
Red italic text highlights security-related properties.

Figure 8.2.2.1-1: The sequence of events when using the Provisioned Symmetric Key Security Association Establishment Framework

Credential Configuration: The Provisioned Secure Connection Key (Kpsa) and the corresponding Provisioned Secure Connection Key Identifier, denoted KpsaID, are provisioned to both entities either with pre-provisioning or remote provisioning. The format of KpsaID is defined in clause 10.5. If Entity A is a CSE, then Entity A shall also be configured with its CSE-ID (not shown in figure 8.2.2.1-1).

Identity Configuration: See clause 8.2.1.

Association Configuration:

- Entity A shall be configured with Entity B identity (IdB) prior to performing the Association Security Handshake. Entity A shall use this identity for Entity B authenticating using the above arguments. This identity is also used to route the (D)TLS exchange. Entity A shall associate Entity B's identity with messages secured within Security Contexts established using the Provisioned Secure Connection Key Kpsa associated with the Provisioned Secure Connection Key Identifier KpsaID.

- If Entity A is a CSE, then Entity B shall be configured with Entity A's CSE-ID prior to performing the Association Security Handshake. If Entity A is an AE, then Entity B may either be configured with Entity A's identity (IdA) prior to performing the Association Security Handshake, or may determine IdA during registration (Creation of the <AE> resource). Entity B shall use this identity for Entity A authenticating using the above arguments. Entity B shall associate the configured Entity A identity with messages secured within Security Contexts established using the Provisioned Secure Connection Key Kpsa associated with the Provisioned Secure Connection Key Identifier KpsaID.

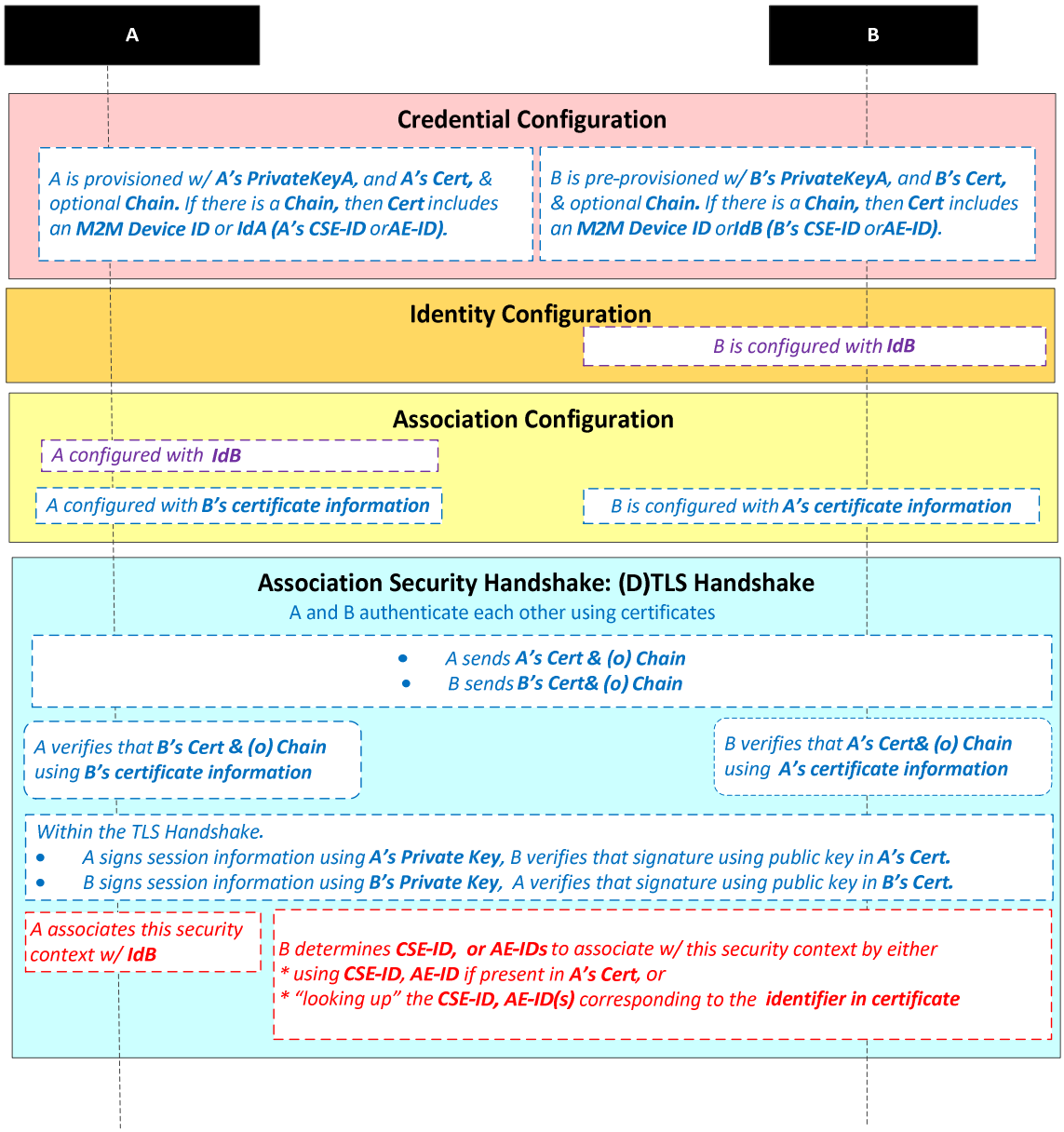
Association Security Handshake: The entities shall perform a (D)TLS-PSK handshake [15] to establish a secure session.

- The "psk_identity" parameter [15] shall be set to the value of the Provisioned Secure Connection Key Identifier KpsaID.
- The entities set the "psk" parameter [15] to the value of the Provisioned Secure Connection Key Kpsa.
- The (D)TLS cipher suite profile for the Provisioned Secure Connection Key Security Association Establishment Framework shall conform to clause 10.2.2.
- Following successful authentication of Entity B, Entity A shall associate the security context with Id B (Entity B's entity identifier) configured to Entity A during Association Configuration.
- Following successful authentication of Entity A, Entity B shall associate the security context with a CSE-ID or AE-ID:
  - If Entity B was already provided with the CSE-ID or AE-ID corresponding to KpsaID, then Entity B shall associate the security context with that CSE-ID or AE-ID.
  - Otherwise, Entity B shall associate the security context with the Credential-ID formed from KpsaID as described in clause 10.4. Entity B shall then determine CSE-ID or AE-ID from the Credential-ID as described in clause 8.2.1.

8.2.2.2 Certificate-Based Security Association Establishment Frameworks

This clause describes the Certificate-Based Security Association Establishment Framework.

Figure 8.2.2.2-1 illustrates the sequence of events when using the Certificate-Based Security Association Establishment Framework. In this description, "Entity A" and "Entity B" correspond to either two CSEs or a CSE and an AE.



NOTE: The following font colours differentiate the general topic that the text relates to:
Blue italic text highlights details specific to this particular Security Association Establishment Framework.
Purple italic text highlights technical actions that may include steps not specified by oneM2M.
Red italic text highlights security-related properties.

Figure 8.2.2.2-1: The sequence of events when using the Certificate-Based Security Association Establishment Framework

Credential Configuration: The private keys and certificates for each entity shall be pre-provisioned as described in clause 8.1.2.3. If Entity A is a CSE, then Entity A shall also be configured with its CSE-ID (not shown in figure 8.2.2.2-1).

Identity Configuration: See clause 8.2.1.

Association Configuration: Entity A and Entity B shall be configured with the information needed for the authentication and identification (during Association Security Handshake) of Entity B and Entity A respectively:

- The information configured to Entity A shall include the following arguments:
  - Entity B's certificate information: as described in clause 8.1.2.4.

- Entity B's identity (IdB). Entity A shall use this identity for Entity B authenticating using the above arguments. This is used to route the (D)TLS exchange.

NOTE: The Entity A will associate Entity B's identity with messages secured within Security Contexts established in accordance with the configured Entity B's certificate information.

- The information configured to Entity B shall include the following argument:
  - Entity A's certificate information: as described in clause 8.1.2.4.

Association Security Handshake:

- Each entity shall verify the other entity's certificate as described in clause 8.1.2.2.
- The entities shall authenticate each other using the validated certificates as specified in TLS 1.2 IETF RFC 5246 [5] and DTLS 1.2 IETF RFC 6347 [6] specifications.
- The (D)TLS cipher suite profile for the Certificate-Based Security Association Establishment Framework shall conform to clause 10.2.3.
- Following successful authentication of Entity B, Entity A shall associate the security context with IdB (Entity B's entity identifier) configured to Entity A during Association Configuration.
- Following successful authentication of Entity A, Entity B shall associate the security context with a CSE-ID, AE-ID or list of allowed AE-IDs:
  - If Entity A establishes a security context by presenting a CSE-ID certificate, then Entity B shall associate the security context with the CSE-ID in the certificate.
  - If Entity A establishes a security context by presenting an AE-ID certificate, then Entity B shall associate the security context with the Absolute AE-ID in the certificate.
  - If Entity A establishes a security context by presenting a Node-ID certificate, then Entity B shall associate the security context with the Credential-ID formed from the globally unique Node identifier in the certificate.
  - If Entity A establishes a security context by presenting a device certificate, then Entity B shall associate the security context with the Credential-ID formed from the globally unique hardware instance identifier in the certificate as described in clause 10.4. Entity B shall then use Credential-ID to determine the CSE-ID, AE-ID or list of allowed AE-IDs as described in clause 8.2.1.
  - If Entity A establishes a security context by presenting a raw public key certificate, then Entity B shall associate the security context with the Credential-ID formed from the corresponding public key identifier described in clause 10.1.2. Entity B shall then use Credential-ID to determine the CSE-ID or AE-ID as described in clause 8.2.1.

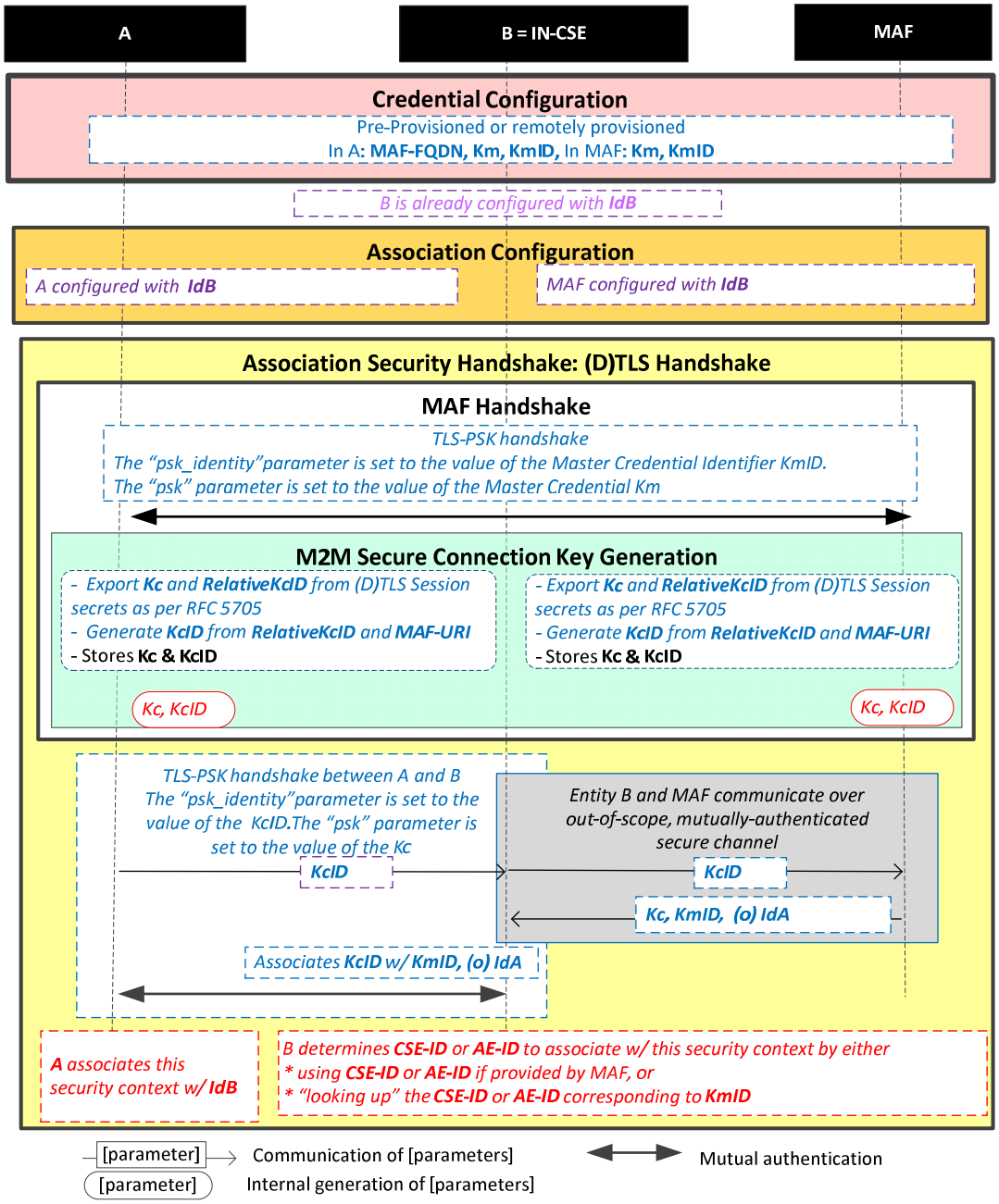
8.2.2.3 MAF-Based Symmetric Key Security Association Establishment Frameworks

This clause describes the MAF-based Security Association Establishment Framework.

This framework uses the MAF Security Framework procedures in clause 8.8, with the following mapping of functional roles:

- Entity A plays the role of the Source End-Point.
- Entity B plays the role of the Target End-Point.

The present clause refers to the entities using only the terminology of Entity A and Entity B.



NOTE: The following font colours differentiate the general topic that the text relates to:
Blue italic text highlights details specific to this particular Security Association Establishment Framework.
Purple italic text highlights technical actions that may include steps not specified by oneM2M.
Red italic text highlights security-related properties.

Figure 8.2.2.3-1: The sequence of events when using the MAF-Based Security Association Establishment Framework

Credential Configuration:

1. Entity A (and Entity B respectively) shall be individually provisioned with credentials for mutual authentication with the MAF, as described in MAF Credential Configuration (clause 8.8.3.1). Pre-provisioning or remote provisioning may be applied. In the case of remote provisioning of symmetric keys, the MAF retrieves the symmetric keys from the MEF during the MAF Client Registration procedure in Identity Configuration.

Identity Configuration:

2. The MAF is expected to be authorized to provide service to Entity A and Entity B.

NOTE 1: The current oneM2M specifications do not describe how this authorization is provided to the MAF.

3. The MAF is configured with information about the identities of Entity B and, optionally, Entity A:

   - If Entity A is a CSE, then the MAF is expected to be configured with Entity A's CSE-ID (denoted IdA).
   - If Entity A is an AE, then the MAF is expected to be configured with Entity A's AE-ID (denoted IdA).
   - The MAF is expected to be configured with Entity B's M2M-SP Assigned CSE-ID (denoted IdB).

NOTE 2: The current oneM2M specifications do not describe how this information is configured to the MAF.

4. If Entity A (or Entity B respectively) are remotely provisioned with a symmetric key for use with the MAF, then Entity A (or Entity B respectively) shall individually perform the MAF Client Registration procedure (clause 8.8.2.3) with the MAF. This procedure is used to trigger the MAF to (a) retrieve Km from the MEF, and (b) provide the End-Point with the KmID to be used for subsequently authentication with the MAF at step 5.

Association Configuration: Entity A and the MAF shall be configured with the information needed for the authentication and identification during MAF Handshake and Association Security Handshake:

5. Authorizing the SAEF:

   - Entity A shall be provided with IdB, the CSE-ID for Entity B. See note 2 in clause 8.2.1.
   - The MAF is expected to be configured with the Entity B Identity (IdB) for which it is authorized to provide Kc for an SAEF with Entity A.

6. Entity A and the MAF shall establish a mutually authenticated secure channel for communication using the MAF Handshake procedure (clause 8.8.2.2), using the credentials provisioned during Credential Configuration.

7. Entity A shall initiate the MAF Key Registration procedure (clause 8.8.2.7) with the MAF. The MAF Key Registration shall include the Security Usage Identifier (SUID) associated with the MAF-Based SAEF and IdB. This procedure results in:

   - Entity A and the MAF establishing a M2M Secure Connection Key (Kc) and associated M2M Secure Connection Key Identifier (KcID), corresponding to the output symmetric key and Key Identifier established by the MAF Key Registration procedure.
   - MAF providing the lifetime for the M2M Secure Connection Key (Kc).

   The SUID limits the scope within which Kc is authorized to be used. In this case, the SUID is used to ensure that the Entity A shall use Kc only with the MAF-Based SAEF.

Association Security Handshake:

8. Entity A shall initiate a (D)TLS-PSK handshake with Entity B, according to clause 10.2.2.

9. Entity A shall send the "Key Identifier" derived from KcID to Entity B (Infrastructure Node) as the "psk_identity" parameter in a (D)TLS-PSK handshake.

10. Entity B recognizes the MAF-FQDN part of the "Key Identifier" in the "psk_identity" parameter, and determines that the corresponding M2M Secure Connection Key (Kc) shall be retrieved from the corresponding MAF. Entity B shall set RelativeKeyID as specified in clause 10.3.5.

11. Entity B and the MAF shall perform the MAF Key Retrieval procedure described in clause 8.8.2.8.

NOTE 3: The MAF Key Retrieval procedure includes establishing a mutually authenticated secure channel for communication using the MAF Handshake procedure (described in clause 8.8.2.2), using the credentials provisioned during Credential Configuration.

Entity B shall provide the RelativeKeyID to the MAF. The MAF returns the output symmetric key value, expirationTime, Security Usage Identifier (SUID), and identity for Entity A to Entity B. The value of Kc shall be set to the output symmetric key value. The Kc Lifetime shall be set to the expirationTime. The SUID limits the scope within which Kc will be used. In this case, the SUID is used to ensure that the Entity B shall use Kc only with the MAF-Based SAEF.

NOTE 4: Assigning Kc Lifetime is the responsibility of the MAF.

12. Entity A and Entity B shall complete the (D)TLS-PSK handshake with the "psk" parameter set to the value of the M2M Secure Connection Key (Kc).

13. Following successful authentication of Entity B, Entity A shall associate the security context with IdB (Entity B's entity identifier) configured to Entity A during Association Configuration.

14. Following successful authentication of Entity A, Entity B shall associate the security context with a CSE-ID or AE-ID:

    - If the MAF provided Entity B with a CSE-ID or AE-ID, then Entity B shall associate the security context with that CSE-ID or AE-ID.

    - Otherwise, Entity B shall associate the security context with the Credential-ID formed from KmID (provided by the MAF) as described in clause 10.4. Entity B shall then determines CSE-ID or AE-ID from the Credential-ID as described in clause 8.2.1.

15. Entity A and Entity B may establish a fresh (D)TLS-PSK handshake using Kc at any time within the Kc Lifetime. Once Kc Lifetime expires, then Entity B shall fail the (D)TLS-PSK handshake, which indicates to Entity B that a fresh MAF Handshake is required.

8.3 Remote Security Provisioning Frameworks

8.3.1 Overview on Remote Security Provisioning Frameworks

8.3.1.1 Purpose of Remote Security Provisioning Frameworks

Remote Security Provisioning Frameworks (RSPFs) provision credentials to an Enrolee, which is a security principal in a Node or CSE or AE, as part of the Enrolment of the Enrolee to an M2M SP or M2M Trust Enabler. The MEF provides its services on behalf of *administrating stakeholders* such as M2M SPs or third party M2M Trust Enablers (MTE). An administrating stakeholder authorizes the MEF Service Provider to provide services to MEF clients, and oversees authorizing the management of credentials.

The credentials are either:

- A symmetric key shared by the Enrolee and an Enrolment Target, which may be a MAF or Node or CSE or AE:

    - If the Enrolment target is an MAF, then the credential can be used for MAF-based SAEF, MAF-based ESPrim and MAF-based ESData Protection Options, with the provisioned symmetric key used for mutual authentication of the Enrolee and the MAF.

    - If the Enrolment target is a Node or CSE or AE, then the credential can be used for only one of PSK-based SAEF or PSK- based ESPrim or PSK-based ESData Protection Options. The provisioned symmetric key used for mutual authentication of the Enrolee and the other Node or CSE or AE.

NOTE 1: This case should be employed only in cases where the Enrolee is expected to require a symmetric key with relatively few CSE or AE.

- Certificate(s) for which the Enrolee knows the corresponding private key, and a set of trust anchors for authenticating the M2M SP or MTE's MAF or other entities enrolled with the M2M SP or MTE. These credentials can be used for:

  - Securing communication directly with other Nodes or CSEs or AEs using Certificate-Based SAEF, Direct End-to-End Key Establishment using Certificates (ESCertKE), and certificate-based ESData protection options. The other Nodes or CSEs or AEs would authenticate themselves using their own certificate(s), chaining to a provisioned trust anchor CA certificate, in these security frameworks.

  - MAF-based SAEF, MAF-based ESPrim, and MAF-based ESData protection options, with the certificate used for authentication of the Enrolee to the MAF. The MAF would authenticate using its own certificate chaining to a provisioned trust anchor CA certificate.

NOTE 2: The RSPFs are specified to provide an interoperable interface for Field Domain entities to interact with an MEF. Use of the specified RSPFs are recommended for use by Field Domain entities because they have been reviewed by the security experts of oneM2M. The RSPFs can also be used by Infrastructure Domain entities (Nodes, AEs, CSEs and MAFs) for interacting with an MEF. It is expected that the MEF may include additional "backend" interfaces, not specified by oneM2M, for coordination of information with administrating stakeholders and MAF Service Providers.

8.3.1.2 High Level Flow

A security principal in a Node or AE or CSE that requires remote provisioning is called an *Enrolee or Source MEF Client.* When a key is being provisioned, then the Nodes or AEs or CSEs or M2M Authentication Function with whom the Enrolee is to establish the symmetric key is called an *Enrolment Target or Target MEF Client.*

The oneM2M system supports the following authentication methods for Remote Security Provisioning Frameworks:

- **Pre-Provisioned Symmetric Enrolee Key Remote Security Provisioning Framework:** A symmetric key is pre-provisioned to the Enrolee and M2M Enrolment Function for the mutual authentication of those entities. For more details, see clause 8.3.2.1.

NOTE 1: The present document supports only pre-provisioned symmetric keys. Future versions intend to add support for authentication using symmetric keys provisioned by other MEF using an RSPF, or other mechanisms.

- **Certificate-Based Remote Security Provisioning Framework:** The Enrolee and M2M Enrolment Function are each issued with:

  - a Private Signing Key that is known only to that entity;

  - a Certificate containing the corresponding Public Verification Key; and

  - (In the case of a device certificate, Node-ID certificate, CSE-ID certificate or AE-ID certificate) a Certificate Chain from the entity's Certificate to a Trust Anchor Certificate.

  The Certificate may be pre-provisioned or provisioned within an RSPF using the Certificate Provisioning procedures specified in clause 8.3.6. If an MEF provisions an MEF Client then the MEF Client shall authenticate itself to the MEF using the latest provisioned certificate from the MEF.

  The Enrolee and M2M Enrolment Function shall validate each other's Certificate before trusting the Public Verification Keys in the Certificate. Within the Security Handshake, the M2M Enrolment Function creates a digital signature of the session parameters using its private signing key and the Enrolee verifies the digital signature using the M2M Enrolment Function's public verification key. Then the roles are reversed: the Enrolee creates a digital signature and the M2M Enrolment Function verifies it.

  For more details see clause 8.3.2.2.

- **GBA-based Remote Security Provisioning Framework**. In this case, the role of the M2M Enrolment Function is performed by a GBA Bootstrap Server Function. This framework uses 3GPP or 3GPP2 symmetric keys to authenticate the Enrolee and the M2M Enrolment Function (which is also a GBA BSF). The details are specified by ETSI TS 133 220 [13] and TIA TIA-1098 [14]. For more details, see clause 8.3.2.3.

The Remote Security Provisioning Frameworks are comprised of the following phases:

- **MEF Client Credential Configuration:** The MEF Client and M2M Enrolment Function are provisioned with the Bootstrap Credential that the entity will use to authenticate itself to the other entity. This phase is also denoted as Bootstrap Credential Configuration.

  - Frequency: If the credential is a symmetric key, then this occurs once per association between the MEF Client and MEF. If the credential is a certificate, then this occurs once per MEF Client.

- **MEF Client Service Configuration:**

  - The MEF Client is configured with the M2M Enrolment Function URI (for the purpose of routing the (D)TLS messages to the M2M Enrolment Function).

  Additionally, in the case of Certificate-Based Remote Security Provisioning Framework:

  - The MEF Client is configured with the M2M Enrolment Function Trust Anchor CA Certificates that the MEF Client will use to verify the M2M Enrolment Function.

  - The M2M Enrolment Function is configured with the MEF Client's certificate information that the M2M Enrolment Function will use to verify the MEF Client's certificate. The necessary certificate information is dependent on the MEF Client's certificate's flavour, with details provided in clause 8.1.2.4.

  - Frequency: This occurs once per association between the MEF Client and MEF.

NOTE 2: In the case of the PPSK RSPF and GBA-Based RSPF, the MEF Client Credential Configuration and MEF Client Assignment would typically occur simultaneously. In the case of the Certificate-based RSPF, the MEF Client Assignment can be separate from MEF Client Credential Configuration.

- **Administrating Stakeholder coordination with MEF** (details are not described in the present document). An Administrating Stakeholder authorizes the MEF to provide services to MEF clients, oversees authorizing the distribution of symmetric keys, and oversees management of security-related MOs on the MEF Client. This typically occurs prior to the MEF Handshake.

  - Frequency: This occurs as requested by the Administrating Stakeholder.

- **Provisioning Procedure Instructions:** The MEF Client either implicitly determines that it is to perform specific provisioning procedures, or is provided with explicit instructions. This triggers the MEF Client to perform the MEF Handshake and initiate provisioning procedures in the Enrolment Exchange.

  - Frequency: This occurs whenever the MEF Client is to initiate a set of provisioning procedures.

  MEF Client Service Configuration, Administrating Stakeholder coordination with MEF, and Provisioning Procedure Instructions phases together are also denoted as Bootstrap Instruction Configuration.

- **MEF Handshake:** Identification, authentication and security context establishment between the MEF Client and M2M Enrolment Function.

  - Frequency: MEF Handshake occurs whenever the MEF client is triggered by Provisioning Procedure Instructions.

  This phase is also denoted as Bootstrap Enrolment Handshake.

- **Enrolment Exchange:**

  - After a successful MEF Handshake in the GBA-Based RSPF, the MEF Client and MEF have established a symmetric key from which keys can be derived for use with other AEs, CSEs or MAF. There is no further interaction between the MEF Client and MEF until the established symmetric key expires, at which point a new handshake is performed.

- After a MEF Handshake in a PPSK-based and Certificate-based RSPFs, the MEF Client and MEF have established a secure channel which is used to protect the Enrolment Exchange used to provision credentials. The Enrolment Exchange is described in more detail in clause 8.3.4. The Enrolment Exchange can include following types of procedures: MEF Client Registration, Symmetric Key Provisioning, Certificate Provisioning, and Device Configuration. The sequence of Enrolment Exchange procedures can be controlled by the MEF Client Command procedure which is outlined in clause 8.3.4 and specified in detail in clause 8.3.9.

- Frequency: This occurs whenever triggered by Provisioning Procedure Instructions.

- **Usage of Provisioned Credentials** The provisioned credentials can then be used in the following types of security frameworks:

  - **Certificate-Based SAEF, ESPrim and ESData:** Certificates and configured trust anchors, are used directly in certificate-based security frameworks with the other Nodes, AEs or CSEs. Trust Anchors can also be configured separately, for example, using ETSI TS 118 122 [57].

  - **PSK-Based SAEF, ESPrim and ESData:** The Source MEF Client and MEF have established a usage-constrained symmetric key, corresponding key identifier and a list of authorized Target MEF Client(s). The Source MEF Client provides the key identifier to Target MEF Client(s) in the security protocol. The Target MEF Client(s) establishes a secure connection to the MEF, and performs the MEF Key Retrieval Procedure (clause 8.3.5.2.8) to retrieve the symmetric key subject to authorization at the MEF.

  - **MAF-Based SAEF, ESPrim and ESData:** If certificates are to be used for authentication to the MAF, then the certificate and trust anchors provisioned during Certificate Enrolment are used for mutual authentication of the MEF Client and MAF. If a symmetric key is used for mutual authentication, then the MEF Client and MEF have established a symmetric key and corresponding key identifier, with constraint for use with a specific MAF. The MEF Client (now acting as a MAF Client) performs the MAF Client Registration procedure, during which the MEF Client/MAF Client provides the key identifier to the MAF. The MAF establishes a secure connection to the MEF, and performs the MEF Key Retrieval Procedure (clause 8.3.5.2.8) to retrieve the symmetric key subject to authorization at the MAF. The MAF provides a KmID for the MEF Client/MAF Client to use in subsequent MAF Handshake procedures.

NOTE 3: If the Enrolment Target hosts a <*ServiceSubscribedAppRule*> resource, then the fetched credentials from the M2M Enrolment Function or the M2M Authentication Function needs to be stored after the Enrolment Target establishes a secured connection with the Enrolee. A Credential ID value in the format as mentioned in clause 10.4 is generated using the credential used for the secured connection establishment and is added into the *applicableCredIDs* attribute of the <*ServiceSubscribedAppRule*> resource.

NOTE 4: If the Enrolee-ID of the Enrolee is retrieved from the M2M Enrolment Function or the M2M Authentication Function, then the same is saved in the *allowedAEs* attribute of the <*ServiceSubscribedAppRule*> resource.

Figure 8.3.1.2-1 illustrates the phases of the Remote Security Provisioning Frameworks.

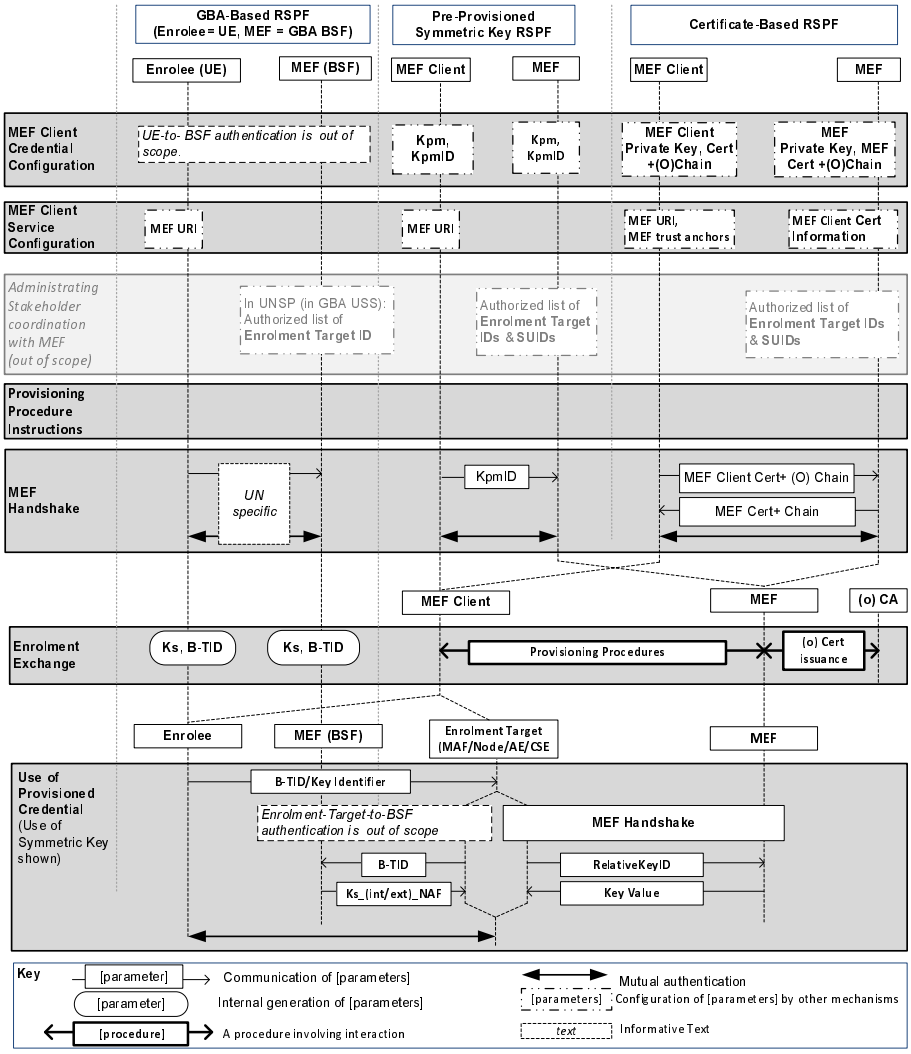


Figure 8.3.1.2-1: Overview of the Remote Security Provisioning Frameworks supported by oneM2M

8.3.2 Detailed Remote Security Provisioning Framework

8.3.2.1 Pre-Provisioned Symmetric Key Remote Security Provisioning Framework

This clause describes the Pre-Provisioned Symmetric Key Remote Security Provisioning Framework. The Bootstrap Credential for this framework is a long-term symmetric key that has been pre-provisioned into the Enrolee and M2M Enrolment Function; this key is called a Pre-Provisioned Symmetric Enrolee Key and is denoted Kpm.

NOTE 1: Long term Pre-Provisioned Symmetric Enrolee Keys can pose a security risk if not adequately secured, and for this reason it is recommended that Long term Pre-Provisioned Symmetric Enrolee Keys are stored in Secure Environments.

Figure 8.3.2.1-1 illustrates the sequence of events when using the Pre-Provisioned Symmetric Enrolee Key Remote Security Provisioning Framework.

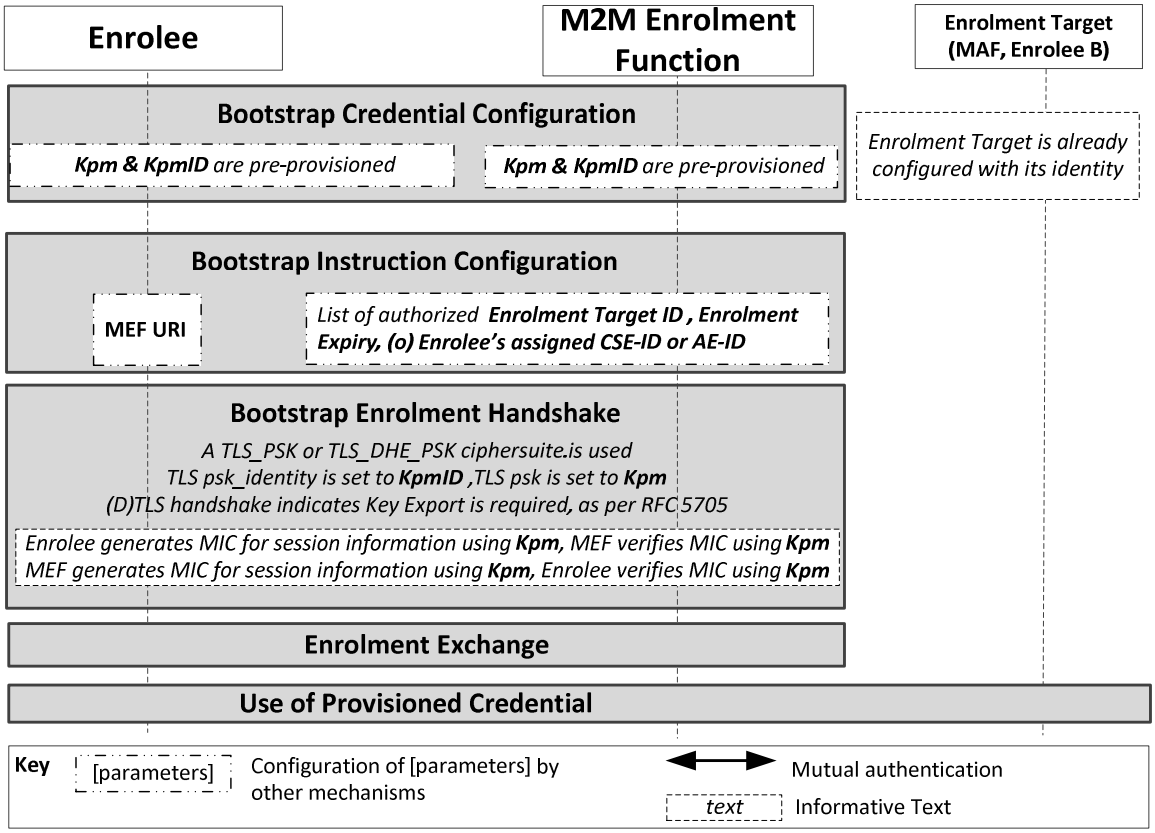


Figure 8.3.2.1-1: The sequence of events when using the Pre-Provisioned Symmetric Key Remote Security Provisioning Framework

Bootstrap Credential Configuration: The Pre-Provisioned Symmetric Enrolee Key (Kpm) and the corresponding Pre-Provisioned Symmetric Enrolee Key Identifier, denoted KpmID, are pre-provisioned to both entities. The Enrolee is also pre-provisioned with the M2M Enrolment Function's URI (MEF URI), for the purpose of routing the (D)TLS exchange.

NOTE 2: This pre-provisioning (by definition) uses mechanisms not specified by oneM2M.

Bootstrap Instruction Configuration: The Enrolee and M2M Enrolment Function are configured with the information needed for authorizing the remote provisioning:

- The Enrolee is configured with (or otherwise obtains) the following arguments to initiate remote provisioning:
  - The Enrolment Target identity: Identifying the Enrolment Target for which the Enrolee is to be provisioned.
  - The Enrolee associates these arguments with the M2M Enrolment Function. The M2M Enrolment Function can be identified to the Enrolee using the Pre-Provisioned Symmetric Enrolee Key Identifier (KpmID) or the M2M Enrolment Function URI.

  Enrolment Expiry: Life Time to be applied for the key generated, i.e. Ke as mentioned in clause 10.7.
- M2M Enrolment Function is configured with the following arguments to authorize the M2M Enrolment Function to remotely provision the Enrolee for an Enrolment Target:
  - The Enrolment Target Identity: Identifying the Enrolment Target for which the Enrolee is to be provisioned.
  - Enrolee's assigned CSE-ID or AE-ID (Enrolee-ID). The M2M Enrolment Function is to provide this entity identity for the Enrolee with the Km or Kpsa to the Enrolment Target, when requested by the Enrolment Target.

- The M2M Enrolment Function associates these arguments with an Enrolee. The Enrolee can be identified to the M2M Enrolment Function using the Pre-Provisioned Symmetric Enrolee Key Identifier (KpmID).

Enrolment Expiry: Life Time to be applied for the keys generated, i.e. Ke. The M2M Enrolment Function may provide this lifetime along with Km or Kpsa to the Enrolment Target.

Bootstrap Security Handshake:

1. The Enrolee and M2M Enrolment Function shall perform a (D)TLS-PSK handshake [15] to establish a secure session.

   - The "psk_identity" parameter [15] is set to the value of the Pre-Provisioned Symmetric Enrolee Key Identifier (KpmID).
   - The "psk" parameter [15] is set to the value of the Pre-Provisioned Symmetric Enrolee Key (Kpm).
   - The (D)TLS cipher suite profile is specified in clause 10.2.2.

Enrolment Key Generation:

2. The Enrolment Key (Ke), RelativeKeID, and Enrolment Re-authentication Key (Ker) are generated from the (D)TLS session secrets by the Enrolee and M2M Enrolment Function using TLS Key Export (IETF RFC 5705 [18]), as described in clause 10.3.1.

3. The Enrolment Key Identifier (KeID) is generated from the RelativeKeID and the M2M Enrolment Function's FQDN by the Enrolee and M2M Enrolment Function, as described in clause 10.3.4.

4. The Enrolee and M2M Enrolment Function store the Enrolment Key (Ke) and Enrolment Key Identifier (KeID), and Enrolment Re-Authentication Key (Ker).

NOTE 3: The Enrolment Key Generation for the Pre-Provisioned Symmetric Enrolee Key Remote Security Provisioning Framework is identical to the Enrolment Key Generation for the Certificate-Based Remote Security Provisioning Framework.

Enrolment Exchange:

5. The Enrolee shall compose a request with a payload containing the parameters and values shown in table 8.3.2.1-1. These parameters could be serialized using, for example, XML or JSON formats.

Table 8.3.2.1-1: Initial request from Enrolee to MEF

Parameter Name	Parameter Value
Certificate Enrolment Indication	<True/False>
MAF Enrolment Indication	<True/False>
Remote Management Indication	<True/False>

6. These indications indicate whether or not the Enrolee is prepared to execute these procedures if instructed by the MEF. The Enrolee shall send the request to the MEF's Enrolment Exchange URI.

7. The MEF shall process the request against the preferences for this Enrolee (see pre-conditions) to determine how the Enrolee is to be provisioned.

NOTE 4: The present document does not define this processing.

8. If the Enrolee does not need to be remotely provisioned for certificate-based authentication with Enrolment Targets, then the MEF shall proceed to step 10. To remotely provision the Enrolee for certificate-based authentication with Enrolment Targets, the MEF shall compose a response with a payload containing the parameters and values shown in table 8.3.2.1-2. These parameters may be serialized using, for example, XML or JSON formats.

Table 8.3.2.1-2: Response from MEF to Enrolee triggering Certificate Enrolment

Parameter Name	Parameter Value
Instruction Type	<Indicating Certificate Enrolment>
URI	<Base certificate enrolment URI>

9. The MEF shall send the response to the Enrolee.

10. If the MEF instructs the Enrolee to perform Certificate Enrolment, then the Enrolee shall perform Certificate Enrolment procedure as described in clause 8.3.6.

11. When Certificate Enrolment is complete, then the Enrolee shall send a request to the MEF indicating success.

12. If the Enrolee does not need to be remotely enrolled with a M2M Authentication Function (MAF), then the MEF shall proceed to step 13. To remotely enrol the Enrolee with a MAF, the MEF shall compose a response with a payload containing the parameters and values shown in table 8.3.2.1-3. These parameters may be serialized using, for example, XML or JSON formats.

Table 8.3.2.1-3: Response from MEF to Enrolee triggering remote enrolment with a MAF

Parameter Name	Parameter Value
Instruction Type	Indicating MAF Enrolment
Credential Type	<Indicating whether to use certificates or symmetric key for authenticating to MAF>
MAF Key Registration URI	<The URI through which MAF Key Registration is performed>
MAF Key Retrieval URI	The URI through which MAF Key Retrieval is performed
(Optional) MAF Client Registration URI	The URI from which the MAF-assigned KmID is retrieved
(Optional) Trust Anchors	Trust anchor CA certificates for MAF certificate
(Optional) Lifetime	Lifetime when the symmetric key shared with MAF will expire

13. The MEF shall send the response to the Enrolee.

14. Upon receipt of this message, the Enrolee shall perform the MAF Client Registration procedure as described in clause 8.8.2.4. This procedure includes the "Use of Remote Provisioned Credential".

15. When the MAF Client Registration procedure is complete, then the Enrolee shall send a message to the MEF indicating success. The MEF may return to step 10, to provision the Enrolee for another MAF.

16. If the Enrolee does not need to be remotely provisioned for remote management server to contact for further configuration, then the MEF shall proceed to step 15. To remotely provision the Enrolee for remote management server, the MEF shall compose a response with a payload containing the parameters and values shown in table 8.3.2.1-4. These parameters may be serialized using, for example, XML or JSON formats.

Table 8.3.2.1-4: Response from MEF to Enrolee provisioning the Enrolee for a remote management server

Parameter Name	Parameter Value
Instruction Type	<Indicating Remote Management Server>
URI	<Base URI of Remote Management Server >

17. The MEF shall send the response to the Enrolee.

18. The Enrolee shall send a message to the MEF acknowledging that it received the instruction. The Enrolee shall initiate contact with the remote management server after the TLS/DTLS session with the MEF is closed.

19. The MEF shall send a response indicating the end of the enrolment exchange.

20. The MEF shall close the TLS/DTLS session.

Use of Remotely Provisioned Credential: In the case where the Enrolment Target is an MAF, the Enrolee is instructed to contact a specific MAF with which to perform Enrolment.

a) If the Enrolee is remotely provisioned with a certificate and trust anchors during the Enrolment Exchange, then the Enrolee may use these in security protocols with the Enrolment Target. Otherwise, the Enrolee shall use the KeID in security protocols with the Enrolment Target as described in the remaining steps.

b) The Enrolee shall provide KeID as a symmetric key identifier in the security protocol.

c) The Enrolment Target checks to see if it has the credentials associated with KeID and if it does not have the credentials, then the Enrolment Target prepares to fetch the credentials from the MEF.

d) The Enrolment Target has been pre-configured with the MEF's FQDN/URL and in order to establish a secured connection, it either uses the PSK credentials or certificate which has also been pre-configured between the Enrolment Target and the MEF. In the case of an Enrolee B, an Enrolment Re-Authentication Key (Ker) established with the MEF may be used for authentication, or a certificate provisioned by the MEF may be used for authentication.

e) If the Enrolment Target wishes to fetch any credential information from the MEF, a retrieve request to MEF with the target URI set to/fetchCredentials/<KeID>/<security-usage-identifier> shall be formed, where <security-usage-identifier> is the SUID for the particular usage of the symmetric key. The originator field of the retrieve request (e.g. X-M2M-Origin header when using HTTPS) contains the ID (AE-ID/CSE-ID/MAF-ID) of the Enrolment Target/MAF.

f) Upon receiving the retrieve request, the MEF performs the following:

i. The MEF extracts the KeID from the Target URI. The MEF shall retrieve the Enrolment Key (Ke) for the corresponding KeID, as per the Enrolment Key Generation process defined in clause 8.3.1.2. If the MEF is unable to retrieve this information, a response shall be sent with an error as per step 'f)vii'.

ii. The Enrolment Target's ID is extracted from the originator field included in the retrieve request, and the Security Usage Identifier (SUID) is extracted from the target URI. The MEF shall validate if the particular Enrolment Target is allowed to fetch credentials for the Enrolee with the particular Security Usage Identifier.

iii. If the validation fails, a response shall be sent as per step 'f)vii'. If the validation is successful, then the key shall be generated with the Enrolment Key (Ke) retrieved in step f)i' as mentioned in clause 10.3.7.

iv. The Enrolee ID corresponding to the KeID is determined by the MEF.

v. The Enrolee Lifetime parameter for the KeID is determined by the MEF. This is a pre-configured value which indicates the validity period of the credentials that are provided to the Enrolment Target/MAF.

vi. A response shall be composed along with a payload containing the parameters and values shown in table 8.3.2.1-5. These parameters may be serialized using, for example, XML or JSON formats.

Table 8.3.2.1-5: Success Response from the MEF to Enrolment Target

Parameter Name	Parameter Value
Status	True
Credential	<Key>
EnroleeID	<Enrolee ID Value>
Lifetime	<Lifetime of generated Key>

vii. Upon any errors in the above steps, the MEF shall compose a response with the parameters shown in table 8.3.2.1-6.

Table 8.3.2.1-6: Failure Response from the MEF to Enrolment Target

Parameter Name	Parameter Value
Status	False
ErrorString	<Failure Reason>

g) The Enrolment Target upon receiving the credentials proceeds to use the credentials in the security protocol with the Enrolee.

8.3.2.2 Certificate-Based Remote Security Provisioning Framework

This clause describes the Certificate-Based Remote Security Provisioning Framework. The Bootstrap Credentials for this framework are Certificates based on asymmetric key pairs.

NOTE 1: Long term asymmetric private keys can pose a security risk if not adequately secured, and for this reason it is recommended that they are stored in Secure Environments. Annex L provides a framework to generate and secure asymmetric key pairs in hardware based Secure Environments.

Figure 8.3.2.2-1 illustrates the sequence of events when using the Certificate-Based Remote Security Provisioning Framework.

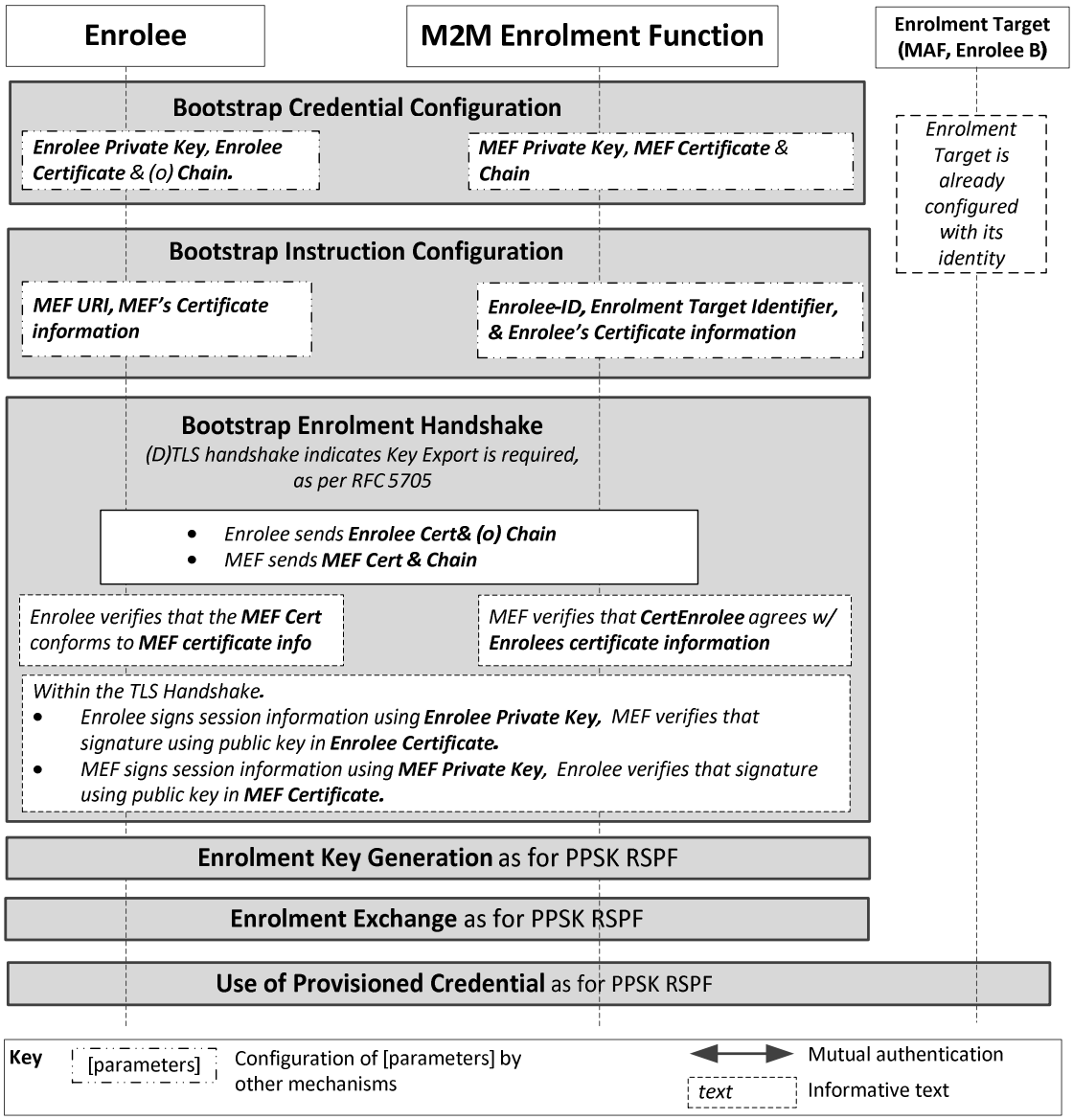


Figure 8.3.2.2-1: The sequence of events when using the Certificate-Based Remote Security Provisioning Framework

Bootstrap Credential Configuration: For this Remote Security Provisioning Framework, Enrolee and M2M Enrolment Function authenticate each other using a Public Key Certificate. The Bootstrap Credentials for the Enrolee and M2M Enrolment Function are pre-provisioned as described in clause 8.1.2.3.

NOTE 2: The identities of the M2M Enrolment Function and Enrolment Target are assumed to have been configured prior to this phase.

Bootstrap Instruction Configuration: In addition to the information identified in clause 8.3.1.2, the Enrolee and M2M Enrolment Function are configured with the information needed for authorizing the remote provisioning:

- The Enrolee is configured with (or otherwise obtains) the following arguments to initiate remote provisioning:
  - Information needed for certificate authentication of the M2M Enrolment Function using an MEF certificate as described in clause 8.1.2.4.
- The M2M Enrolment Function is configured with the following arguments describing Enrolee authorized to perform Security Handshake with M2M Enrolment Function:
  - Information needed for certificate authentication of the Enrolee, as described in clause 8.1.2.4.

Bootstrap Security Handshake: The Enrolee and M2M Enrolment Function perform a (D)TLS handshake as specified in TLS 1.2 IETF RFC 5246 [5] and DTLS 1.2 IETF RFC 6347 [6] specifications.to establish a secure session.

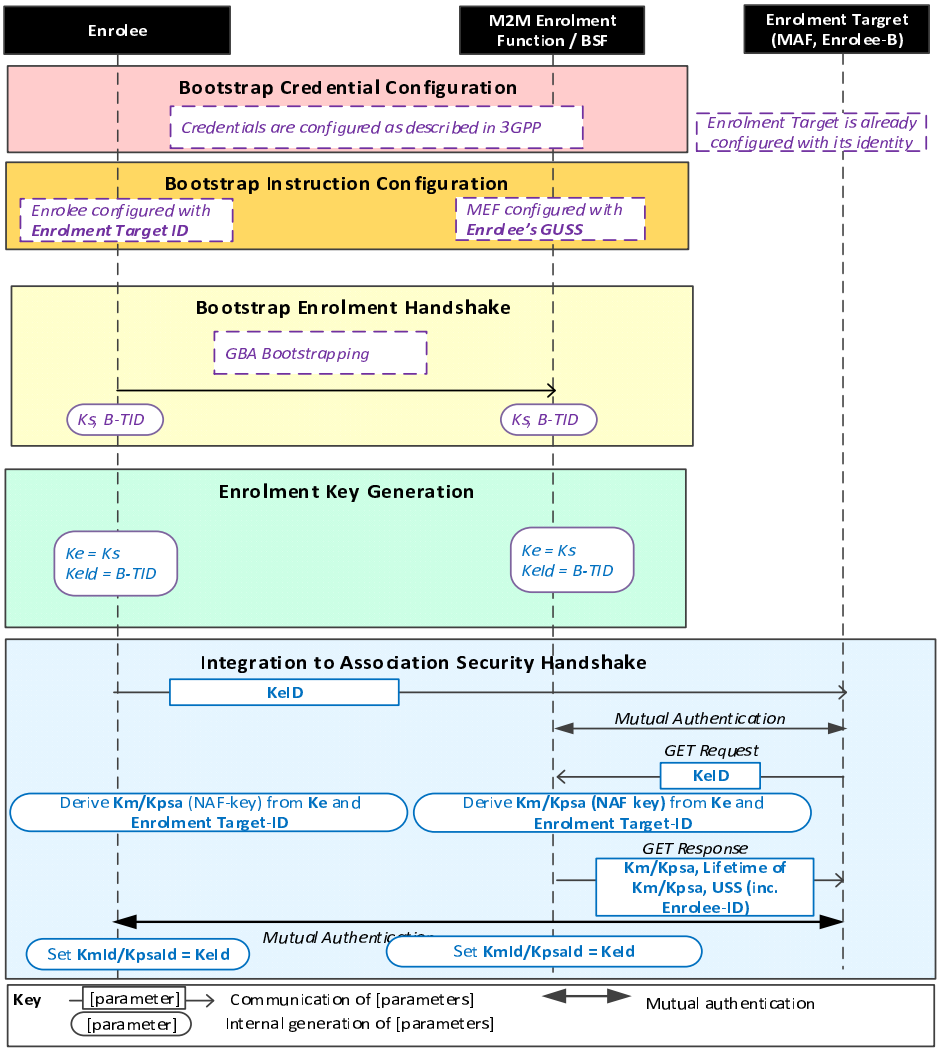
- Each entity (Enrolee and M2M Enrolment Function) verifies the other entity's certificate as described in clause 8.1.2.5.
- The Enrolee and M2M Enrolment Function authenticate each other using the validated certificates as specified in TLS 1.2 IETF RFC 5246 [5] and DTLS 1.2 IETF RFC 6347 [6] specifications.
- The (D)TLS cipher suite profile is specified in clause 10.2.3.

Enrolment Key Generation, Enrolment Exchange and **Use of Provisioned Credentials:**

- The steps are identical to those shown for "Enrolment Key Generation", "Enrolment Exchange" and "Use of Provisioned Credentials" in clause 8.3.2.1.

8.3.2.3 GBA-Based Remote Security Provisioning Framework

To share a long term Master Credential (Km) or Provisioned Secure Connection Key (Kpsa) between an Application Service/Middle Node and an Enrolment Target, the M2M Application Service/Middle Node shall perform a successful GBA bootstrapping and derive a NAF key (Ks_(ext/int)_NAF). This NAF key is the Master Credential (Km) or Provisioned Secure Connection Key (Kpsa).



NOTE: The following font colours differentiate the general topic that the text relates to:
Black text contains Remote Security Provisioning -Framework-independent details.
Blue italic text highlights details specific to this particular Remote Security Provisioning Framework.
Purple italic text highlights technical actions that may include steps not specified by oneM2M.

Figure 8.3.2.3-1: The sequence of events when using the GBA-Based Remote Security Provisioning Framework

Bootstrap Credential Configuration: The credentials configuration for Enrolee and M2M Enrolment Function (MEF) is described in ETSI TS 133 220 [13]. The MEF plays the role of the BSF. The credentials used to perform mutual authentication between Enrolee and MEF are UNSP specific.

Bootstrap Instruction Configuration: The Enrolee, the MEF and the Enrolment Target shall be configured with the information needed for authorizing the remote provisioning:

- The Enrolee shall be configured with the Enrolment Target Identity: identifying the Enrolment Target for which the Enrolee is to be provisioned.
- The MEF shall be configured with the Enrolee-ID and the Enrolment Target Identity:
  - The Enrolment Target Identity: Identifying the Enrolment Target for which the Enrolee (authenticated using the GBA) is to be provisioned.
  - The Enrolee's assigned CSE-ID or AE-ID (Enrolee-ID), The M2M Enrolment Function is to provide this entity identity for the Enrolee with the Km or Kpsa to the Enrolment Target, when requested by the Enrolment Target.

- Enrolee's GBA User Security Settings (GUSS) enables indicating if Enrolee is allowed to establish a NAF-specific key with the Enrolment Target or/and if the BSF can distribute a NAF specific key to the Enrolment Target.

Bootstrap Enrolment Handshake:

The Bootstrap Enrolment Handshake enables the establishment of a GBA bootstrapped key (Ks) shared between the Enrolee and the MEF with associated Bootstrapping Transaction Identifier (B-TID) and key lifetime, by performing to the GBA Bootstrapping phase described in ETSI TS 133 220 [13].

If a bootstrapped key Ks is already shared between Enrolee and the MEF and still valid, then the Bootstrap Enrolment Handshake phase is not needed. The Enrolment Key Generation phase can take place with the existing GBA Bootstrapped key Ks.

Enrolment Key Generation phase:

The Enrolment Key (Ke) shall be the GBA Bootstrapped key (Ks) established during the Bootstrap Enrolment Handshake.

The Enrolment Key Identifier (Ke-ID) shall be the Bootstrapping Transaction Identifier (B-TID) generated during the Bootstrap Enrolment Handshake.

Integration to the Association Security Handshake:

- The Enrolee and the Enrolment Target shall establish the Master Credential (Km) or the Provisioned Secure Connection Key (Kpsa) thanks to procedures described in ETSI TS 133 220 [13] using the Enrolment Key (Ke) as GBA bootstrapped key Ks and the Enrolment Key Identifier (Ke-ID) as B-TID. The Enrolment Target plays the role of a NAF.

  - The Enrolee and the Enrolment Target shall establish NAF-specific key(s) as described in ETSI TS 133 220 [13]. A key lifetime is associated to the NAF-specific keys. The Enrolment Target also receives the Enrolee's User Security Settings (USS) from the MEF/BSF:

    - The FQDN of the NAF, used as input to generate the Ks_(int/ext)_NAF, shall be set as follows:

      - In the case where the Enrolment Target is an M2M Authentication Function, then the FQDN of the NAF is set to the FQDN of the M2M Authentication Function.

      - In the case where the Enrolment Target is a CSE, then the FQDN of the NAF is set to the public domain name representation of the CSE-ID as defined in ETSI TS 118 101 [1].

    - In case of GBA_ME, NAF-specific key is Ks_NAF.

    - In case of GBA_U, NAF-specific keys are Ks_int_NAF and Ks_ext_NAF.

  - The Master Credential (Km)) or the Provisioned Secure Connection Key (Kpsa) shall be the NAF-specific key:

    - In case of GBA_ME, Km/Kpsa = Ks_NAF.

    - In case of GBA_U, Km/Kpsa = Ks_int_NAF if HTTP Client application resides in the UICC. Otherwise, Km/Kpsa = Ks_ext_NAF.

  - The Enrolee and the Enrolment Target shall set the Master Credential Identifier (KmID) or the Provisioned Secure Connection Key Identifier (KpsaID) to the value of KeID.

Enrolee and Enrolment Target shall perform (D)TLS-PSK handshake (IETF RFC 4279 [15]) with the Master Credential (Km) or Provisioned Secure Connection Key (Kpsa) as Pre-Shared Key in compliance with clause 10.2.2. If UICC is used as Secure Environment supporting Remote Security Provisioning, GBA-U with Km/Kpsa = Ks_int_NAF shall be used for authentication and key exchange.

8.3.3 Void

8.3.4 Enrolment Exchange

8.3.4.1 Enrolment Exchange Procedures

The following procedures may occur within an Enrolment Exchange:

- MEF Client Registration procedures.
- Symmetric Key Provisioning procedures.
- Certificate Provisioning procedure.
- Device Configuration procedures, per ETSI TS 118 122 [57] can be applied, with the MEF interacting with a DM Server and MEF Client interacting with the DM Client on the Managed Entity.
- MEF Client Command Procedures (i.e. CRUD procedures targeting at a <*mefClientCmd*> resource), which enable the MEF to control the sequence of Enrolment Exchange procedures.

The clauses below describe triggering mechanisms specific to each set of procedures. Alternatively, other mechanisms, not specified by oneM2M, can be used to trigger any Enrolment Exchange procedure, with the condition that such mechanisms provide a satisfactory level of security. Example mechanisms include pre-configuration and manual configuration.

8.3.4.2 MEF Client Registration

MEF Client Registration procedures are specified in clauses 8.3.5.2.3, 8.3.5.2.4, 8.3.5.2.5 and 8.3.5.2.6.

MEF Client Registration procedures can only be performed within an Enrolment Exchange.

MEF Client Registration procedures can be triggered by the following oneM2M-specified mechanisms:

- **Procedures triggered using Device Configuration.** Device Configuration, specified in ETSI TS 118 122 [57], can trigger MEF Registration Procedures:
  - Adding a [*MEFClientRegCfg*] MO triggers the MEF Client to perform the MEF Client Registration Procedure, specified in clause 8.3.5.2.3.
  - Deleting a [*MEFClientRegCfg*] MO triggers the MEF Client to stop using the associated MEF Client registration, delete any credentials associated with that MEF Client registration and end the associated MEF Client registration on the MEF. The MEF achieves the final step by performing the MEF Client De-Registration Procedure, specified in clause 8.3.5.2.6.

8.3.4.3 Symmetric Key Provisioning

Symmetric Key Provisioning procedures are specified in clauses 8.3.5.2.7, 8.3.5.2.8, 8.3.5.2.9 and 8.3.5.2.10.

These procedures can only be performed within an Enrolment Exchange.

These procedures can be triggered by the following oneM2M-specified mechanisms:

- **Procedures triggered using a "MO_Node" MEF Client Command:** Device Provisioning (ETSI TS 118 122 [57]) can be used to configure MEF Client with an [*authenticationProfile*] MO which has a child [*MEFClientRegCfg*] MO node to instruct the MEF Client that Symmetric Key Provisioning will be used for credentials used in that [*authenticationProfile*] MO. If a MEF Client receives of an "MO_NODE" MEF Client Command matching the path of such a [*authenticationProfile*] MO, then this can trigger a Symmetric Key Provisioning procedure according to the information elements in the MEF Client Command and the current values of the parameters in these MO nodes.

NOTE: Using Device Configuration to update or delete the [*authenticationProfile*] MO and/or its child [*MEFClientRegCfg*] MO node does not implicitly trigger a Symmetric Key Provisioning procedure. The update or delete will not take effect until a Symmetric Key Provisioning procedure is trigger by some other mechanisms.

- **Procedures triggered by an expiry of a MEF Key Registration.** If the MEF Client previously (successfully) executed an MEF Key Registration procedure under the control of an [*authenticationProfile*] MO on the MEF Client, and the current time is greater than the *expirationTime* of the [*authenticationProfile*] resource, and if the current time is close to or greater than the *expirationTime* of the most recent MEF Key Registration, then this can trigger the MEF Client to perform MEF Key Registration. The criteria for being "close to the *expirationTime*" is left up to the implementation of the MEF Client.

- **Procedures triggered by receiving, within a oneM2M security protocol, a symmetric key identifier whose FQDN matches the MEF's FQDN**. If a Target MEF Client receives, within a oneM2M security protocol, a symmetric key identifier whose FQDN matches the MEF's FQDN, then this can trigger the Target MEF Client to execute the MEF Key Retrieval Procedure specified in clause 8.3.5.2.8. See steps 6 and 7 in clause 8.3.5.1.

8.3.4.4 Certificate Provisioning

Certificate Provisioning procedures are specified in clause 8.3.6.

These procedures can only be performed within an Enrolment Exchange.

These procedures can be triggered by the following oneM2M-specified mechanisms:

- **Procedures triggering using MEF Client Command Procedure:** If the MEF Client receives a MEF Client Command identifying a Certificate Provisioning Procedure, then this triggers the MEF Client to execute the Certificate Provisioning procedure using the information elements included in the command.

8.3.4.5 Device Configuration

Device Configuration is specified in ETSI TS 118 122 [57].

Device Configuration can be performed within an Enrolment Exchange with a MEF, or in a DM session with other DM servers (separate from an Enrolment Exchange). Clause 8.3.8 specifies use of Device Configuration within an Enrolment Exchange with a MEF.

Device Configuration can be triggered by the following oneM2M-specified mechanisms:

- **Procedures triggered using MEF Client Command Procedure**: If the MEF Client receives a MEF Client Command identifying the Device Configuration Procedure, then this trigger the MEF Client to execute a Device Configuration session using the information elements included in the command.

8.3.4.6 MEF Client Command

MEF Client Command procedures are specified in clause 8.3.9.

MEF Client Command procedures can only be performed within an Enrolment Exchange.

MEF Client Command procedures can be triggered by the following oneM2M-specified mechanisms:

- **Procedures triggered following MEF Client Registration Procedure**

  - A MEF Client Command Retrieve shall be executed following an MEF Client Registration procedure (other than MEF Client De-registration).

- **Procedures triggered according to *retryDuration***

  - When the MEF issues a NO_MORE_COMMANDS MEF Client Command, then the *cmdArgs* includes a *retryDuration* providing the duration after which the MEF Client attempts MEF Client Command Retrieve. A *retryDuration* is cancelled whenever the MEF Client successfully interacts with the MEF prior to this time. For further details see clause.8.3.9.6.

- **Procedures triggered following an attempt to perform an issued MEF Client Command**

  - If the MEF Client has attempted executing a previously issued MEF Client Command, then the MEF Client shall perform the MEF Client Command Update procedure to report on the status of that execution. The MEF can issue a MEF Client Command in the response.

    An Example of a MEF Client Command procedure is illustrated in figure 8.3.4.6-1.

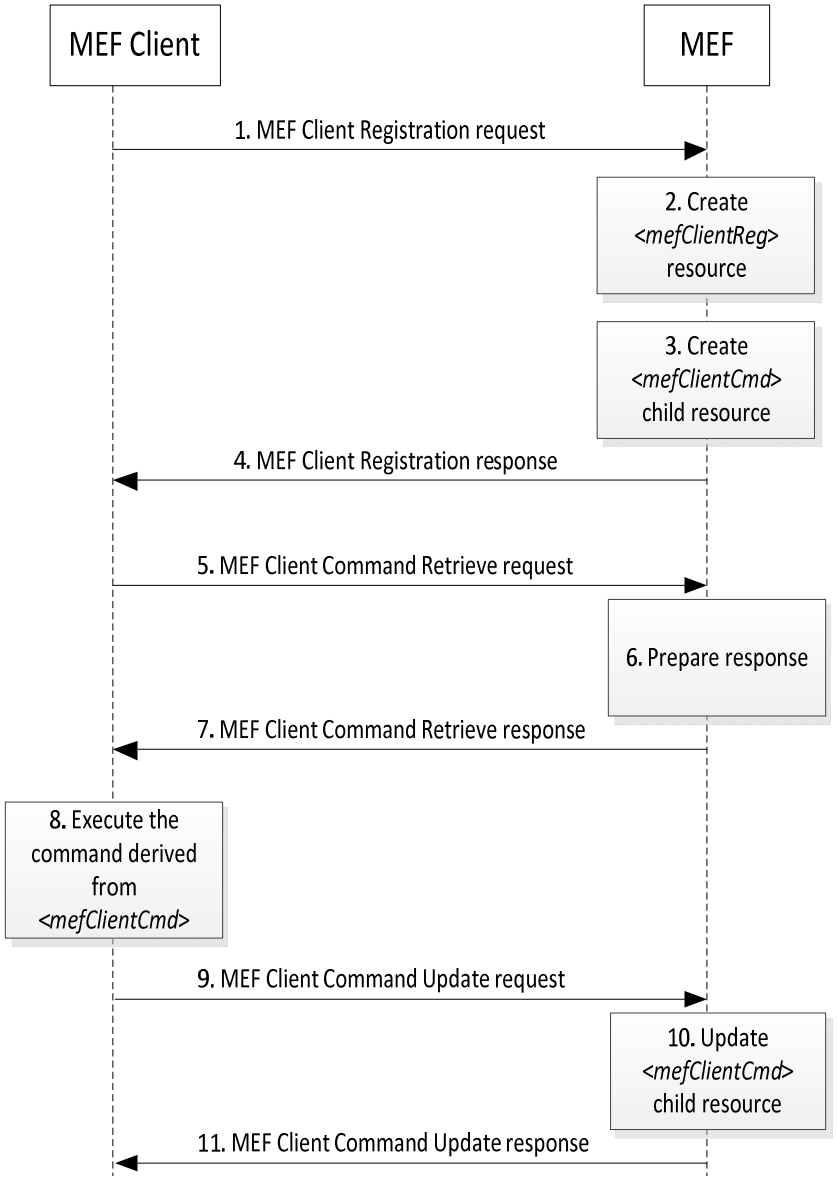


Figure 8.3.4.6-1: Example MEF Client Command procedure

1) The MEF client sends an MEF Client Registration request.

2) The MEF creates a <*mefClientReg*> resource.

3) If the MEF wants to issue a MEF Client Command it creates a <*mefClientCmd*> resource as child of the <mefClientReg> resource.

4) The MEF sends the MEF Client Registration response which includes a representation of the <*mefClientReg*> resource, including the *childResource* reference, whose value represents the resource ID of a <*mefClientCmd*> resource.

5) The presence of the *childResource* reference triggers the MEF client to retrieve the <*mefClientCmd*> resource. The MEF Client sends a MEF Client Command Retrieve request to the MEF.

6) The MEF forms the response.

7) The MEF returns a MEF Client Command Retrieve response which includes the <*mefClientCmd*> resource.

8) The MEF client parses the received response and executes the command included therein.

9) After execution of the command, the MEF client reports the result to the MEF by a MEF Client Command Update Request.

10) The MEF updates the <*mefClientCmd*>. If the MEF has a new command for the MEF Client it indicates a trigger in the representation of the <*mefClientCmd*> resource.

11) The MEF sends the MEF Client Command Update Response. If the received response includes another MEF Client Command, steps 8 to 11 are repeated.

8.3.5 Symmetric Key Provisioning Details

8.3.5.1 Introduction

Clause 8.3.5 describes the common details and procedures for using an RSPF to provision symmetric keys.

These frameworks use a MEF to provide authentication and distribution of symmetric key for use by a Source End-Point initiating establishing the symmetric key, and one or more Target End-Points. Table 8.3.5.1-1 MEF Clients can retrieve the output symmetric key from the MEF. The MEF provides its services on behalf of administrating stakeholders such as M2M SPs or third party M2M Trust Enablers (MTE). An administrating stakeholder authorizes the MEF to provide services to MEF clients, and oversees authorizing the distribution of symmetric keys. Table 8.3.5.1-1 describes the mapping of Source MEF Client and Target MEF Client to roles in the specific MEF-Based Frameworks, and the allowed number of Target MEF Clients.

Table 8.3.5.1-1: Mapping to specific Security Frameworks

MEF-Based Security Framework	Source MEF Client	Target MEF Client	Number of Target MEF Clients	Output Symmetric Key
MAF Security Frameworks	MAF Client	MAF	1	M2M Master Key (Km)
Security Association Establishment Framework (SAEF)	Entity A	Entity B	1	M2M Secure Connection Key (Kc)
End-to-End Security of Primitives (ESPrim)	Originator	Receiver	1	pairwiseESPrimKey
End-to-End Security of Data (ESData)	Source ESData End-Point	Target ESData End-Point	1..n	ESData Key

This clause 8.3.5 specifies *MEF Procedures* between the MEF Clients and associated messages. The operation and management of the MEF, beyond the details provided for the MEF Procedures, are not specified in the present document.

The general sequence for using the MEF procedures is shown in figure 8.3.5.1-1 and described as follows:

1. Each MEF Client shall separately establish credentials for mutual authentication with the MEF as described in **MEF Client Credential Configuration** (clause 8.3.7.1).

2. Each MEF client shall be separately configured to register on the MEF with a specific administrating stakeholder. **MEF Client Registration Configuration** (clause 8.3.7.2) provides the necessary parameters.

3. Each MEF Client shall perform a **MEF Client Registration procedure** with the MEF. This provides confirmation that the MEF Client is willing to use the services of the MEF, under the authorization of the administrating stakeholder. The MEF client shall register separately for each administrating stakeholder, even when registering via a single MEF. If the MEF Client is remotely provisioned for mutual authentication with the MEF, then the MEF shall provide the MEF Client with the KmID to be used for subsequently authentication with the MEF.

At a later time independent of this sequence of events, the **MEF Client Registration Update procedure** may be performed to confirm that the MEF Client is willing to use the services of the MEF and or establish a new Km and KmID, and the **MEF Client De- Registration procedure** may be performed to signal that the MEF Client is ceasing use the services of the MEF.

4. The Source MEF client shall be configured to establish secure communication using a security feature (SAEF, ESPrim or ESData) with symmetric keys established via the MEF. The details of this configuration is specific to the security feature being invoked, but shall include the **MEF Key Registration Configuration** (clause 8.4.4.3).

5. The Source MEF Client shall perform a **MEF Key Registration procedure** to establish a symmetric key and corresponding identifier. The Source MEF Client shall also provide the Security Usage Identifier (SUID) limiting the scope of the credential by identifying the security feature (SAEF, ESPrim or ESData). This procedure shall include the **MEF Handshake procedure** for mutual authentication of the Source MEF Client and MEF.

   At a later time independent of this sequence of events, the **MEF Key Registration Update procedure** may be performed to update the expiration of the registered key or update the list of Target MEF Clients, and the **MEF Key De-Registration procedure** may be performed to delete the key registration from the MEF.

6. The Source MEF Client shall provide, to the Target MEF Client(s), the symmetric key identifier established in the MEF Key Registration procedure. The details of this step depend on the security feature as identified by the SUID.

7. The Target MEF Client shall perform the **MEF Key Retrieval procedure**, to retrieve the symmetric key and corresponding information. This procedure shall include the **MEF Handshake** procedure for mutual authentication of the Target MEF Client and MEF.

8. The symmetric key shall be used in the security protocol between the Source MEF Client and Target MEF Client. If the security protocol requires a single symmetric key, then the first half of the distributed symmetric key shall be used. If the security protocol requires two symmetric keys (for example, an encryption key and a separate integrity key), then the two halves of the distributed symmetric key shall be used as the two security protocol symmetric keys. The details of this step depend on the security feature.

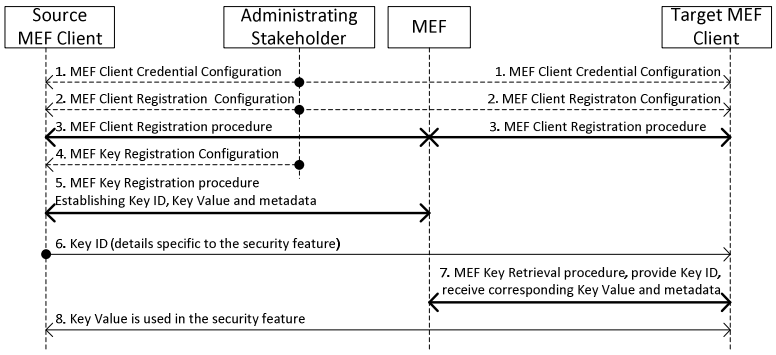


Figure 8.3.5.1-1: The sequence of events when using the MEF Security Framework as part of a security feature

Clause 8.3.5.2 describes the processing and information flows of the MEF Procedures. Clause 8.3.7 describes the information in the MEF Client Credential Configuration, MEF Client Registration Configuration and MEF Key Registration Configuration.

8.3.5.2 MEF Security Framework Processing and Information Flows

8.3.5.2.1 Introduction

Clause 8.3.5.2 specifies the processing and information flows of the MEF procedures.

8.3.5.2.2 MEF Handshake Procedure

Purpose: A MEF Handshake procedure establishes a mutually authenticated TLS or DTLS session for protecting the communication between an MEF Client and MEF. In the case of the MEF Key Registration procedure, the TLS or DTLS session may be used by the Source MEF Client and MEF to establish the Key Value.

Pre-Conditions: One of the following conditions shall hold:

- The MEF Client and MEF have been provisioned with certificates as described in the MEF Client Credential Configuration details in clause 8.8.3.1, and configured with CA certificates for validating certificates as described in the MEF Client Registration Configuration details in clause 8.8.3.2.
- The MEF Client and MEF have established a symmetric Master Credential (Kpm) with corresponding Master Credential Identifier (KpmID). The Kpm and KpmID may be pre-provisioned, or Kpm may be established using Remote Security Provisioning Framework with KpmID established using the MEF Client Registration procedure.

NOTE: In the case of establishing Kpm via remote provisioning, MEF Handshake cannot be performed during MEF Client Registration because (a) the MEF does not know Kpm prior to MEF Client Registration and (b) KpmID has not been assigned prior to MEF Client Registration.

Procedure description:

- If the MEF Client and MEF have established a symmetric Master Credential (Kpm) with corresponding Master Credential Identifier (KpmID), then the MEF Client and MEF shall establish the TLS or DTLS session using the TLS-PSK handshake according to clause 10.2.2, with the following details:
  - The "psk_identity" parameter [15] shall be set to the value of the Master Credential Identifier (KpmID).
  - The "psk" parameter [15] shall be set to the value of the Master Credential (Kpm).
- If the MEF Client and MEF are to authenticate using certificates, then the MEF Client and MEF shall establish the TLS or DTLS session using the certificate-based TLS handshake according to clause 10.2.2, with the following details:
  - The TLS server certificate shall be the MEF's certificate. The MEF Client shall verify the MEF's certificate against the set of provisioned MEF certificate trust anchors as described in clause 8.1.2.5.
  - The TLS client certificate shall be the 'MEF Client's certificate. The MEF shall verify the 'MEF Client's certificate against the provisioned MEF Client Certificate Information as described in clause 8.1.2.5.

8.3.5.2.3 MEF Client Registration Procedure

Purpose: The MEF Client registers with the MEF to confirm that it is willing to use the services of the MEF, under the authorization of the administrating stakeholder.

NOTE: The MEF Client Registration procedure is equivalent to CSE or AE registration, but in this case the MEF Client is "registering" to the MEF, and not the registrar CSE.

Pre-Conditions: The MEF Client, MEF, and (where applicable) MEF have been provisioned with the parameters described in clause 8.3.7.

Procedure description:

1) The MEF Client shall establish a TLS (or DTLS) connection with the MEF by performing the MEF Handshake procedure (clause 8.3.5.2.2). This provides the MEF with an authenticated identity for the MEF Client.

2) The MEF Client shall send a MEF Client Registration request including the information shown in table 8.3.5.2.3-1.

Table 8.3.5.2.3-1: MEF Client Registration Request message information

Parameter	Description	Multiplicity
MEF-FQDN	FQDN of the MEF, from MEF Instruction Configuration	1
expirationTime	Proposed time when the registration shall expire.	1
Labels	Labels to aid discovery the record of the MEF Client's registration.	0..1
adminFQDN	FQDN of the administrating stakeholder, provided in the MEF Client Registration Configuration	1

3) Upon receiving the request, the MEF shall process the request. If error cases are encountered, then the MEF shall send an error response. The MEF may assign different values for parameters received from the MEF Client, based on instruction from the administrating stakeholder. If the request is processed successfully, then the MEF shall compose a MEF Client Registration response request including the information shown in table 8.3.5.2.3-2.

Table 8.3.5.2.3-2: MEF Client Registration Response message information

Parameter	Description	Multiplicity
MEFClientRegID	An identifier for the new MEF Client Registration record.	1
labels	Labels to aid discovery of the MEF Client Registration record	0..1
expirationTime	Time when the MEF Client Registration record shall expire.	1
MEF Client ID	Identifier of the MEF Client	1
adminFQDN	FQDN of the administrating stakeholder	1

The MEF shall send the response to the MEF Client.

4) The MEF Client and MEF shall store the parameters.

8.3.5.2.4 MEF Client Configuration Retrieval Procedure

Purpose: This procedure enables a MEF Client to retrieve MEF Client Configurations provided by the administrating stakeholder to the MEF.

Pre-Conditions:

- The MEF Client has previously performed the MEF Client Registration procedure to create the MEF Client Registration record.
- The MEF Client Registration record is not expired.

Procedure Description. The procedure comprises the following steps:

1. The MEF Client shall establish a TLS (or DTLS) connection with the MEF as described in step 1 of clause 8.3.5.2.3.
2. The MEF Client shall send a MEF Client Configuration Retrieval request including the information shown in table 8.3.5.2.4-1.

Table 8.3.5.2.4-1: MEF Client Configuration Retrieval Request message information

Parameter	Description	Multiplicity
MEF-FQDN	FQDN of the MEF, from MEF Instruction Configuration	1
MEFClientRegID	Identifier for the MEF Client registration record being updated	1

3. Upon receiving the request, the MEF shall process the request. If error cases are encountered, including if there is no MEF Client Configuration currently associated with the identified MEF Client registration record, then the MEF shall send an error response. If the request is processed successfully, then the MEF shall attempt to retrieve the MEF Client Configuration currently associated with the identified MEF Client registration record.

4. The MEF shall compose a MEF Client Configuration Retrieval response a containing the following parameters.

Table 8.3.5.2.4-2: MEF Client Configuration Retrieval Response message information

Parameter	Description	Multiplicity
MEFClientCfg	MEF Client Configuration currently associated with the identified MEF Client registration record	1

The MEF shall send the response to the MEF Client.

5. The MEF Client shall apply the MEF Client Configuration.

8.3.5.2.5 MEF Client Registration Update Procedure

Purpose: This procedure enables a MEF Client to update the MEF Client registration by any combination of extending the *expirationTime* of the MEF Client Registration record or updating the *labels*.

Pre-Conditions:

- The MEF Client has previously performed the MEF Client Registration procedure to create the MEF Client Registration record.
- The MEF Client Registration record is not expired.

Procedure Description. The procedure comprises the following steps:

1. The MEF Client shall establish a TLS (or DTLS) connection with the MEF as described in step 1 of clause 8.3.5.2.3.
2. The MEF Client shall send a MEF Client Registration Update request including the information shown in table 8.3.5.2.5-1.

Table 8.3.5.2.5-1: MEF Client Registration Update Request message information

Parameter	Description	Multiplicity
MEF-FQDN	FQDN of the MEF, from MEF Instruction Configuration	1
MEFClientRegID	Identifier for the MEF Client registration record being updated	1
expirationTime	Proposed time when the MEF Client registration record shall expire	0..1
labels	Labels to aid discovery of the MEF Client registration record	0..1
NOTE: At least one of *expirationTime* and labels shall be included.		

3. Upon receiving the request, the MEF shall process the request. If error cases are encountered, then the MEF shall send an error response. If the request is processed successfully, then the MEF shall update the MEF Client Registration record with the proposed values if authorized by the administrating stakeholder. The MEF may assign different values for parameters received from the MEF Client, based on instruction from the administrating stakeholder.
4. The MEF shall compose a MEF Client Registration Update response containing the following parameters.

Table 8.3.5.2.5-2: MEF Client Registration Update Response message information

Parameter	Description	Multiplicity
expirationTime	Updated time when the MEF Client Registration record shall expire.	0..1
labels	Updated labels to aid discovery of the MEF Client Registration record	0..1
NOTE: The response only includes *expirationTime* and/or *labels* if those parameters were present in the corresponding request.		

The MEF shall send the response to the MEF Client.

5. The MEF Client and MEF shall store the parameters.

8.3.5.2.6 MEF Client De-Registration Procedure

Purpose: This procedure enables a MEF Client to end its registration with the MEF.

Pre-Conditions:

- The MEF Client has previously performed the MEF Client Registration procedure to create the MEF Client Registration record.
- The MEF Client Registration record is not expired.

Procedure Description. The procedure comprises the following steps:

1. The MEF Client shall establish a TLS (or DTLS) connection with the MEF as described in step 1 of clause 8.3.5.2.3.
2. The MEF Client shall send a MEF Client De-Registration request including the information shown in table 8.3.5.2.6-1.

Table 8.3.5.2.6-1: MEF Client De-Registration Request message information

Parameter	Description	Multiplicity
MEF-FQDN	FQDN of the MEF, from MEF Instruction Configuration	1
MEFClientRegID	Identifier for the MEF Client Registration record being ended	1

3. Upon receiving the request, the MEF shall process the request. If error cases are encountered, then the MEF shall send an error response. If the request is processed successfully, then the MEF shall delete the information associated with the identified MEF Client Registration record.
4. The MEF shall compose a MEF Client Registration Update response indicating the success of the operation. The MEF shall send the response to the MEF Client.

8.3.5.2.7 MEF Key Registration Procedure

Purpose: This procedure enables a Source MEF Client to establish a symmetric key with the MEF which can be retrieved for use by one or more Target MEF Clients.

This procedure is performed between the Source MEF Client and the MEF.

Pre-Conditions:

- The Source MEF Client is provided with (or has otherwise determined) the information in the MEF Key Registration Configuration (clause 8.3.7.3).
- The Source MEF Client has performed the MEF Client Registration procedure (clause 8.3.5.2.3) with the MEF for the administrating stakeholder identified in the MEF Key Registration Configuration.

Procedure Description. The procedure comprises the following steps:

1. The Source MEF Client shall establish a TLS or DTLS session with the MEF using the MEF Handshake procedure, described in clause 8.3.5.2.2. A by-product of the MEF Handshake procedure is that the MEF establishes an authenticated identity for the Source MEF Client.
2. The Source MEF Client selects the value of the M2M Secure Connection Key (Kc) to be distributed by the MEF. The value shall be one of the following:
   - The Source MEF Client generates the output symmetric key value from the (D)TLS session secrets using TLS Key Export (IETF RFC 5705 [18]), as described in clause 10.3.1.

- The output symmetric key value is self-generated by the Source MEF Client, independently of the (D)TLS session secrets.

3. The Source MEF Client shall compose a list of Target MEF Clients to whom the MEF is authorized to provide the output symmetric key value:

   - In the case of MEF-Based SAEF or MEF-Based ESPrim: The list shall contain exactly one Absolute AE-ID or Absolute CSE-ID.

   - In the case of MEF-Based ESData: The list shall contain any non-zero number of Absolute AE-ID or Absolute CSE-IDs.

NOTE 1: How the Source MEF Client selects the list of Target MEF Clients is application dependent.

4. The Source MEF Client shall send a MEF Key Registration request, including the information shown in table 8.3.5.2.7-1.

Table 8.3.5.2.7-1: MEF Key Registration Request message information

Parameter	Description	Multiplicity
MEF-FQDN	FQDN of the MEF, from MEF Instruction Configuration	1
expirationTime	Proposed time when the Key Registration shall expire.	1
labels	Labels to aid discovery of the Key Registration	0..1
adminFQDN	Identifier for the administrating stakeholder	1
SUID	The Security Usage Identifier limiting the security feature in which the symmetric key may be used.	1
targetIDs	(Optional) list of identifiers for the initial set of Target MEF Clients authorized to retrieve the symmetric key.	0..1
Key Value	(Optional) If present, this parameter contains an output symmetric key value which is self-generated by the Source MEF Client. If this parameter is not present, then the Source MEF Client and MEF will generate the output symmetric key value using TLS Exporter	0..1

5. The MEF shall process the request. If error cases are encountered, then the MEF shall send an error response. If the request is processed successfully, then the MEF shall authorize establishing a Key Value, based on the authenticated identity for the Source MEF Client.

NOTE 2: The present document provides no details for the authorization of this request.

6. If the request included a value in the Key Value parameter, then the MEF shall store this value. Otherwise, the MEF shall generate Key Value from the (D)TLS session using TLS Key Export (IETF RFC 5705 [18]), as described in clause 10.3.1.

7. The MEF shall initialize the list of authorized Target MEF Clients (those MEF Clients which may to retrieve this credential) to the list provided in the request.

   - In the case of MEF-Based ESData: This list may be further updated by administrating stakeholders during or after the MEF Key Registration procedure.

NOTE 3: The present document does not provide any details about administrating stakeholders updating the list of authorized Target MEF Clients on the MEF. The MEF could provide its own logic and interface allowing administrating stakeholders to manage this list.

8. The MEF shall select a previously-unused value of RelativeKeyID.

9. The MEF may assign different values for parameters received from the MEF Client, based on instruction from the administrating stakeholder.

10. The MEF shall send a response, to the Source MEF Client, including the information shown in table 8.3.5.2.7-2.

Table 8.3.5.2.7-2: MEF Key Registration response message information

Parameter	Description	Multiplicity
RelativeKeyID	The relative part of the Key Identifier associated with the Key Registration	1
expirationTime	Time when the Key Registration shall expire.	1
Source MEF Client ID	Identifier of the Source MEF Client	1
labels	Labels to aid discovery of the Key Registration	0..1
adminFQDN	Identifier for the administrating stakeholder	1
SUID	The Security Usage Identifier limiting the security feature in which the symmetric key may be used.	1
targetIDs	List of identifiers for the initial set of Target MEF Clients authorized to retrieve the symmetric key. This list may have been modified from the list provided by the MEF Client, or created by the MEF (if the MEF Client did not provide a list).	1

11. The Source MEF Client and MEF shall store the output symmetric key value and corresponding Key Identifier.

    - The Key Identifier is generated from the RelativeKeyID and the M2M Authentication Function's FQDN by the Source MEF Client and MEF, as described in clause 10.3.5.

8.3.5.2.8 MEF Key Retrieval Procedure

Purpose: This procedure enables a Target MEF Client to retrieve the Key Value from a MEF corresponding to a RelativeKeyID received by the Target MEF Client.

Pre-Conditions:

- The Target MEF Client has performed the MEF Client Credential Configuration (clause 8.3.5.2.1) with the MEF, including configuration of the MEF Key Retrieval URI.
- The Source MEF Client has performed the MEF Key Registration procedure (clause 8.3.5.2.2) with the MEF, resulting in a registered Key Value and assigned RelativeKeyID for a specific administrating stakeholder and Security Usage Identifier (SUID).
- The Target MEF Client received a Key Identifier from the Initiating-MEF Client in a security feature with the SUID which the Source MEF Client provided to the MEF during the MEF Key Registration procedure (clause 8.3.5.2.7). The Key Identifier shall be composed of the FQDN of the MEF and the RelativeKeyID assigned to the registered key.
- The Target MEF Client may expect that it is authorized to obtain the corresponding output symmetric key value.

NOTE: The Target MEF Client should not repeat this procedure if the Target MEF Client is already in possession of the corresponding Key Value.

Procedure Description. The procedure comprises the following steps:

1. The Target MEF Client shall establish a TLS or DTLS session with the MEF using the MEF Handshake procedure, described in clause 8.3.5.2.2. A by-product of the MEF Handshake procedures is that the MEF establishes an authenticated identity for the Target MEF Client.
2. The Target MEF Client shall send a MEF Key Retrieval request to the MEF including the information shown in table 8.3.5.2.8-1.

Table 8.3.5.2.8-1: MEF Key Retrieval Request message information

Parameter	Description	Multiplicity
RelativeKeyID	The relative part of the Key Identifier received from the Source MEF Client in a security feature	1

3. The MEF shall process the request. If error cases are encountered, then the MEF shall send an error response. If the request is processed successfully, then the MEF shall identify the key registration using the *RelativeKeyID*.

4. The MEF shall determine if the Target MEF Client is authorized to retrieve the registered key and metadata by comparing the authenticated identifier for Target MEF Client against the list of identifiers for authorized Target MEF Clients. If the Target MEF Client is not authorized, then the MEF shall send, to the Target MEF Client, an error message. Otherwise, the MEF shall proceed to the next step.

5. The MEF shall send a response, to the Target MEF Client, including the information shown in table 8.3.5.2.8-2.

Table 8.3.5.2.8-2: MEF Key Registration response message information

Parameter	Description	Multiplicity
expirationTime	Time when the Key Registration shall expire.	1
Source MEF Client ID	Identifier of the Source MEF Client	1
labels	Labels to aid discovery of the Key Registration	0..1
adminFQDN	Identifier for the administrating stakeholder	1
SUID	The Security Usage Identifier limiting the security feature in which the symmetric key may be used.	1
Key Value	The registered value of the output symmetric key	1

6. The Target MEF Client shall associate the parameters with the key identifier.

8.3.5.2.9 MEF Key Registration Update Procedure

Purpose: This procedure enables a Source MEF Client to update the metadata associated with a registered key.

This procedure is performed between the Source MEF Client and the MEF.

Pre-Conditions:

- The MEF Client has previously performed the MEF Key Registration procedure to create the key registration.
- The key registration is not expired.

Procedure Description. The procedure comprises the following steps:

1. The MEF Client shall establish a TLS (or DTLS) connection with the MEF as described in step 1 of clause 8.3.5.2.7.

2. The Source MEF Client shall compose a list of Target MEF Clients to whom the MEF is authorized to provide Kc:

   - In the case of MEF-Based SAEF or MEF-Based ESPrim: The list shall contain exactly one Absolute AE-ID or Absolute CSE-ID.

   - In the case of MEF-Based ESData: The list shall contain any non-zero number of Absolute AE-ID or Absolute CSE-IDs.

NOTE 1: The present document does not provide any details about how the Source MEF Client selects the list of Target MEF Clients.

3. The Source MEF Client shall send a MEF Key Registration Update request, including the updated information shown in table 8.3.5.2.9-1.

Table 8.3.5.2.9-1: MEF Key Registration Update Request message information

Parameter	Description	Multiplicity
MEF-FQDN	FQDN of the MEF, from MEF Instruction Configuration	1
RelativeKeyID	The relative part of the Key Identifier associated with the Key Registration	1
expirationTime	Proposed time when the Key Registration shall expire.	0..1
labels	Proposed Labels to aid discovery of the registered key	0..1
targetIDs	(Optional) proposed list of identifiers for the set of Target MEF Clients authorized to retrieve the symmetric key	0..1
NOTE: At least one of expirationTime, labels or targetIDs shall be provided.		

4. The MEF shall process the request. If error cases are encountered, then the MEF shall send an appropriate error response. If the request is processed successfully, then the MEF shall update the metadata with the proposed values if authorized by the administrating stakeholder. The MEF may assign different values for parameters received from the MEF Client, based on instruction from the administrating stakeholder.

5. The MEF shall send a response, to the Source MEF Client, including the information shown in table 8.3.5.2.9-2.

Table 8.3.5.2.9-2: MEF Key Registration Update response message information

Parameter	Description	Multiplicity
expirationTime	Current time when the key registration shall expire, if changed since the last time the MEF Client was provided with the expiration time.	0..1
labels	Updated list of labels to aid discovery of the Key Registration, if any.	0..1
targetIDs	Current list of identifiers for the initial set of Target MEF Clients authorized to retrieve the symmetric key. This list may have been modified from the list provided by the MEF Client.	0..1
NOTE: The response includes only those parameters that were present in the corresponding request.		

8.3.5.2.10 MEF Key De-Registration Procedure

Purpose: This procedure enables a Source MEF Client to request the MEF to stop distributing the registered key.

This procedure is performed between the Source MEF Client and the MEF.

Pre-Conditions:

- The MEF Client has previously performed the MEF Key Registration procedure to create the key registration.
- The key registration is not expired.

Procedure Description. The procedure comprises the following steps:

1. The MEF Client shall establish a TLS (or DTLS) connection with the MEF as described in step 1 of clause 8.3.5.2.7.

2. The MEF Client shall send MEF Key De-Registration request including the information shown in table 8.3.5.2.10-1.

Table 8.3.5.2.10-1: MEF Client De-Registration Request message information

Parameter	Description	Multiplicity
MEF-FQDN	FQDN of the MEF, from MEF Instruction Configuration	1
RelativeKeyID	The relative part of the Key Identifier associated with the Key Registration	1

3. Upon receiving the request, the MEF shall process the request. If error cases are encountered, then the MEF shall send an error response. If the request is processed successfully, then the MEF shall delete the information associated with the identified key registration.

4. The MEF shall compose MEF Client De-Registration response indicating the success of the operation. The MEF shall send the response to the MEF Client.

8.3.5.3 Mapping to Protocol in ETSI TS 118 132

The Mmef Interface defined in ETSI TS 118 132 [58] shall be used for symmetric key provisioning procedures. The mapping of the MEF Procedures described in clause 8.3.5.2 to the Mmef interface is described in ETSI TS 118 132 [58].

8.3.6 Certificate Provisioning Procedure Details

8.3.6.1 Introduction

The Certificate Provisioning procedure includes the following actors:

- MEF Client: a Security Principal requesting provisioning of an MEF-Provisioned Certificate. The MEF Client uses the MEF-Provisioned Certificate for subsequent authentication of itself to the MEF. The Security Principal can use the MEF-Provisioned Certificate for subsequent authentication of itself in other oneM2M Security Principals.

- MEF CA: issuing MEF-Provisioned Certificates.

- MEF: serving requests from the MEF Client, and acting as a Registration Authority (RA) to forward Certificate Signing Requests (CSRs) towards the MEF CA. The MEF can request the MEF CA to add attributes to those attributes already present in the CSR, and can request deletion or modification of attributes present in the CSR.

The Certificate Provisioning Procedure only specifies the interaction between the MEF Client and the MEF.

NOTE 1: The present document does not describe the interaction between the MEF and MEF CA.

The Certificate Provisioning Procedure achieve the following outcomes:

- The MEF Client obtains MEF-Provisioned Certificate.

- The MEF Client obtains the MEF CA's Certificate(s). This certificate(s) shall be used by the MEF Client for subsequent validation of certificates authenticating the MEF. This certificate(s) may be used by the Security Principal for subsequent validation of certificates authenticating other Security Principals and MAFs.

NOTE 2: Additional trust anchor CA certificates for validation of other Security Principals and MAFs can also be provisioned by configuration of MOs based on the [*trustAnchorCred*] resource.

The Certificate Provisioning Procedure comprises two procedures:

- Initial Certificate Provisioning Procedure: used when the MEF Client does not possess a valid MEF-Provisioned Certificate that was previously provisioned by the MEF.

- Certificate Re- Provisioning Procedure: used by a MEF Client to renew/rekey its existing valid MEF-Provisioned Certificate that was previously provisioned by the MEF.

The present document describes use of the following protocols for the Certificate Provisioning Procedure:

- Enrolment over Secure Transport (EST), specified in IETF RFC 7030 [59]. The use of this protocol is described in clause 8.3.6.2.

- Certificate Provisioning functions using Simple Certificate Enrolment Protocol (SCEP) [66]. The use of this protocol is described in clause 8.3.6.3.

8.3.6.2 Certificate Provisioning procedures using EST

8.3.6.2.1 Introduction

The Enrolment over Secure Transport (EST) protocol is specified in IETF RFC 7030 [59]. When EST is used for Certificate Provisioning procedures, then the following mapping of concepts shall be applied.

- The MEF Client acts as the EST Client.
- The MEF acts as the EST Server.
- The MEF CA acts as the EST CA.
- The MEF-Provisioned Certificate is equivalent to the EST Client Certificate.

If a MEF or MEF Client claiming support of the Certificate Provisioning Procedure using EST, then:

- The MEF or MEF Client shall support the mandatory EST operations and the optional "CSR Attributes" operations (see figure 5 of IETF RFC 7030 [59]).
- The MEF or MEF Client shall support TLS server authentication with certificates and TLS client authentication with certificates as specified for EST in sections 3.3.1 and 3.3.2 of IETF RFC 7030 [59].

NOTE 1: This is used when a Certificate-based RSPF is used. The Certificate-based RSPF mandates that the MEF Client and MEF use only trust anchor CA certificates which have been explicitly identified for use for validating MEF Certificates and MEF Client Certificates. These correspond to the Explicit Trust Anchors (TAs) as defined in section 1.1 of [59]. Consequently, the MEF Client/EST Client uses an EST Client Explicit TA database, and the MEF/EST Client uses an EST Client Explicit TA database, where these databases are defined in figure 4 of [59].

- If the MEF or MEF Client supports PPSK-based RSPF, then the MEF or MEF Client shall support EST Certificate-Less TLS Mutual Authentication (section 3.3.3 of IETF RFC 7030 [59]).
- The MEF or MEF Client may support linking identity and Proof-of-Possession information (section 3.5 of IETF RFC 7030 [59]).

NOTE 2: Until widely-used cryptographic libraries are available which support this functionality, it is unlikely that this functionality would be supported by the MEF or MEF Client.

- The MEF or MEF Client shall not use the HTTP-based client authentication feature of EST (section 3.2.3 of [59]).

NOTE 3: HTTP-based client authentication in EST can be used in scenarios where the MEF Client is authorized using user authentication as discussed in section 2.2.3 of IETF RFC 7030 [59]. These scenarios have not yet been considered by the present document. These scenarios can be supported in the future by adding support for HTTP-based client authentication.

- The MEF Client shall support generation of private/public key pairs. The MEF Client and MEF shall use server-side key generation feature of EST (sections 2.4 and 4.4 of IETF RFC 7030 [59]).

8.3.6.2.2 Initial Certificate Provisioning procedure using EST

Purpose: Enabling an MEF Client to request its first certificate from the MEF. See also the initial enrolment operational scenarios in section 2.2 of IETF RFC 7030 [59], noting the supported authentication methods listed in clause 8.3.6.2.1.

Pre-Conditions:

A. Common Pre-conditions for all Certificate Provisioning Procedures:

   i. The MEF Client and MEF support EST.

   ii. The MEF Client is provided with the estBaseURI whose FQDN shall match the FQDN of the MEF.

   iii. The MEF Client is triggered to perform EST.

NOTE 1: The estBaseURI in pre-condition A.ii can be provided in the message triggering EST in pre-condition A.iii.

B. The MEF Client and MEF have successfully performed a MEF Handshake and the MEF associates an identifier with the MEF Client. For the Initial Certificate Provisioning procedure, one of the following RSPFs shall be used:

   i. PPSK-Based RSPF (clause 8.3.2.1) corresponding to certificate-less TLS authentication in EST, described in section 3.3.3 of [59].

   ii. Certificate-Based RSPF (clause 8.3.2.2) corresponding to mutual, certificate-based TLS authentication in EST:

      1) The certificate used to authenticate the MEF/EST Server corresponds to the EST Server certificate (defined in figure 3 of [59]) which the MEF Client/ EST Client validates against the EST Client Explicit Trust Anchor database (see note 1 in clause 8.3.6.2.1). EST describes the EST Server authentication in section 3.3.1 of [59], which mandates the EST Client perform EST Server authorization checks in section 3.6 of [59] with details specific to authorization checks for an EST Client Explicit TA database in section 3.6.1 of [59].

      2) The certificate used to authenticate the MEF Client/EST Client corresponds to a Third-Party EST Client certificate, (defined in figure 3 of [59]) which the MEF/ EST Server validates against the EST Server Explicit Trust Anchor database (see note 1 in clause 8.3.6.2.1). EST describes the EST Client authentication in section 3.3.2 of [59], which mandates the EST Server perform authorization checks in section 3.7 of [59].

NOTE 2: HTTP-based client authentication of the user or EST Client is not supported by the Initial Certificate Provisioning procedure. See note 3 in clause 8.3.6.2.1.

Procedure Description:

1. Obtaining trust anchor CA certificates. See section 4.1 of [59].

   a. The MEF Client shall request the set of trust anchor CA certificates as described in section 4.1.2 of [59].

   b. The MEF shall respond a set of trust anchor CA certificates as described in section 4.1.3 of [59].

   c. The MEF Client is expected to install the trust anchor CA certificates.

NOTE 3: Certification path validation and certificate status verification needs to be performed by the MEF Client as specified in clause 8.1.2.2.

2. Obtaining the set of CSR attributes. See section 4.5 of [59].

   a. The MEF Client shall request the set of CSR attributes from the MEF as described in section 4.5.1 of [59].

   b. The MEF shall respond with the set of required CSR attributes as described in section 4.5.2 of [59]. The set of required CSR attributes shall comply with the CSR Profile in clause 10.1.4. This set includes a challengePassword and an identity attribute for the type of identifier which the MEF associates with the MEF Client (see pre-conditions).

3. Obtaining a Certificate.

   a. The MEF Client shall either generate a public/private key pair of suitable key length, or select an existing public/private key pair of suitable key length.

   b. The MEF Client shall generate a CSR with the requested CSR attributes using the key pair.

   c. The MEF Client shall request a EST Client certificate using "Simple Enrolment of Clients" as described in section 4.2.1 of IETF RFC 7030 IETF RFC 7030 [59].

   d. The MEF shall validate the attributes, including the challengePassword, against those provided in step 2. The MEF shall validate the identity attribute against the authenticated identity associated with the MEF Client (see precondition B).

NOTE 4: The MEF, acting as a Registration Authority (RA), forwards the CSR to a Certificate Authority (CA). The CA issues the EST Client certificate (defined in figure 3 of IETF RFC 7030 [59]) and returns the certificate to the MEF.

e. The MEF shall send the EST Client certificate (defined in figure 3 of IETF RFC 7030 [59]) to the MEF Client in the response as described in section 4.2.3 of IETF RFC 7030 [59].

f. The MEF Client is expected to install the EST Client certificate and associates it with the corresponding private key. The EST Client certificate shall be used for subsequent authentication with the MEF. The EST Client certificate may also be used as an end-entity certificate in other security protocols.

8.3.6.2.3 Certificate Re-Provisioning procedure using EST

Purpose: Enabling an MEF Client to renew/rekey a currently valid Enrolled Certificate. See also the client certificate reissuance operational scenario in section 2.3 of IETF RFC 7030 [59].

Pre-Conditions: [59**]**

A. Common Pre-conditions for all Certificate Provisioning Procedures: see pre-condition A in clause 8.3.6.2.2.

B. The MEF Client has previously performed the Initial Certificate Provisioning procedure or Certificate Re-Provisioning Procedure with the MEF, and the MEF Client has installed its EST Client certificate and EST Client Explicit Trust Anchor database from the most recent such procedure.

C. The MEF Client and MEF have performed a MEF Handshake for the Certificate-Based RSPF (clause 8.3.2.2) with the MEF Client using its EST Client certificate and EST Client Explicit Trust Anchor database as discussed in precondition B. The details are identical to pre-condition B.ii in clause 8.3.6.2.2, with the difference that the MEF Client/EST Client authenticates itself with an EST Client Certificate (defined in figure 4 of [59]) rather than the EST Third-Party EST Client certificate in pre-condition B.ii.2.

Procedure Description:

1) Obtaining trust anchor CA certificates. As for step 1 in clause 8.3.6.2.2.

2) Obtaining the set of CSR attributes. As for step 2 in clause 8.3.6.2.2.

3) Obtaining a Certificate. As for step 3 in clause 8.3.6.2.2, except step 3.c and 3.d are replaced with the following:

4) The MEF Client shall request the renewal/rekeying of its EST Client certificate using "Simple Re-Enrolment of Clients" as described in section 4.2.2 of [59].

5) The MEF shall validate the challengePassword and EKU(s) against those provided in step 2. The MEF validates the provided identity against the identity associated with the MEF Client (see precondition C).

8.3.6.3 Certificate Provisioning procedures using SCEP

8.3.6.3.1 Introduction

The Simple Certificate Enrolment Protocol (SCEP) is specified in the IETF RFC 8894 [66].

When SCEP is used for Certificate Provisioning procedures, the following mapping of concepts shall be applied:

- The M2M Enrolment Function (MEF) Client acts as the SCEP Client.
- The MEF acts as the SCEP Server (also known as a SCEP Responder).
- The MEF CA acts as the SCEP CA.
- The MEF-Provisioned Certificate is equivalent to the SCEP Client Certificate.

If a MEF or MEF Client claim support of the Certificate Provisioning Procedure using SCEP, then:

- The MEF or MEF Client may support linking identity and Proof-of-Possession information.

NOTE 1: Until widely-used cryptographic libraries are available which support this functionality, it is unlikely that this functionality would be supported by the MEF or MEF Client.

- The MEF or MEF Client shall not use the HTTP-based client authentication.

NOTE 2: HTTP-based client authentication in SCEP can be used in scenarios where the MEF Client is authorized using user authentication. These scenarios have not yet been considered by the present document. These scenarios can be supported in the future by adding support for HTTP-based client authentication.

- The MEF Client shall support generation of private/public key pairs.

8.3.6.3.2 Details of Certificate Provisioning procedures using SCEP

Purpose: Enabling an MEF Client to request its first certificate and subsequent certificates from the MEF using SCEP as specified in IETF RFC 8894 [66].

Pre-Conditions:

A. Common Pre-conditions for all Certificate Provisioning Procedures:

   i) The MEF Client and MEF support SCEP.

   ii) The MEF Client is provided with the base URI whose FQDN shall match the FQDN of the MEF.

   iii) The MEF Client is triggered to perform SCEP.

NOTE 1: The base URI in pre-condition A.ii can be provided in the MEF Client command triggering SCEP in pre-condition A.iii.

B. If the client does not have an appropriate existing certificate then it shall generate locally a self-signed certificate. The keyUsage extension in the certificate shall indicate that it is valid for digitalSignature and keyEncipherment. The self-signed certificate should use the same subject name as in the PKCS #10 request. See section 2.4 of IETF RFC 8894 [66].

Procedure Description:

1. Obtaining trust anchor CA certificates. See section 4.2 of [66].

   a) The MEF Client shall request the set of trust anchor CA certificates.

   b) The MEF shall respond a set of trust anchor CA certificates.

   c) The MEF Client is expected to install the trust anchor CA certificates.

NOTE 2: Certification path validation and certificate status verification needs to be performed by the MEF Client as specified in clause 8.1.2.2.

2. Obtaining the CA capabilities. See section 3.5 of IETF RFC 8894 [66].

   a) The MEF Client shall request the set of CA capabilities from the MEF as described in section 3.5.1 of [66].

   b) The MEF shall respond with the set of required CSR attributes as described in section 3.5.2 of IETF RFC 8894 [66].

3. Obtaining a Certificate.

   a) The MEF Client shall either generate a self-signed public/private key pair of suitable key length, or select an existing public/private key pair of suitable key length.

   b) The MEF Client shall generate a CSR with the requested CSR attributes using the key pair.

   c) The MEF Client shall request a SCEP Client certificate using "Certificate Enrolment" as described in section 4.3 of IETF RFC 8894 [66].

   d) The MEF shall validate the attributes, optionally also a challengePassword. The MEF shall validate the identity attribute against the authenticated identity associated with the MEF Client.

NOTE 3: The MEF, acting as a Registration Authority (RA), forwards the CSR to a Certificate Authority (CA). The CA issues the SCEP Client certificate and returns the certificate to the MEF.

e) The MEF shall send the SCEP Client certificate to the MEF Client in the response as described in section 4.3.1 of IETF RFC 8894 [66].

f) The MEF Client is expected to install the SCEP Client certificate and associates it with the corresponding private key. The SCEP Client certificate shall be used for subsequent authentication with the MEF. The SCEP Client certificate may also be used as an end-entity certificate in other security protocols.

8.3.7 MEF Client Configuration Details

8.3.7.1 MEF Client Credential Configuration Details

The MEF Client and MEF shall be configured with credentials for mutual authentication of the MEF Client and MEF.

The credentials for mutual authentication shall be either pre-provisioned or remotely provisioned by another MEF using Remote Security Provisioning Frameworks, or by Device Configuration as specified in ETSI TS 118 122 [57]. Either symmetric key credentials or certificate credentials maybe provisioned. Symmetric key credentials may be used for authenticating some MEF Clients and certificate credentials may be used for authenticating other MEF Clients. The selection may be based on the capabilities of the MEF Client.

The details depend on the type of credential (symmetric key or certificates) and, in the case of symmetric keys, the type of provisioning (pre-provisioning or remote provisioning).

1) Details specific to **Pre-Provisioned Symmetric Keys (PPSKs)**: the Pre-Provisioned Symmetric Enrolee Key (Kpm) and corresponding key Identifier (KpmID) shall be provisioned to the MEF Client (assuming the role of Enrolee) and the MEF.

2) Details specific to **Remotely-Provisioned Symmetric Keys (RPSKs)**: The MEF Client and an M2M Enrolment Function (MEF) shall be provisioned with credentials for performing a Remote Security Provisioning (RSPF) Framework. The MEF Client shall be authorized to use the services of the MEF. For more details, see clause 8.3.2.

NOTE 1: In this case, the Pre-Provisioned Symmetric Enrolee Key (Kpm) and key Identifier (KpmID) are established during the MEF Client Registration procedure.

3) Details specific to **Certificates (whether pre-provisioned or remotely provisioned)**: The MEF Client shall be provisioned with an MEF Client certificate with optional certificate chain. The MEF Client certificate shall be a device certificate, Node-ID certificate, AE-ID certificate or CSE-ID certificate.

NOTE 2: The configuration of MEF trust anchor CA certificates is addressed in MEF Client Registration Configuration, and can occur separately from MEF Client Credential Configuration.

The oneM2M Device Configuration specification ETSI TS 118 122 [57] provides a set of <*mgmtObj*> specializations that shall be used for MEF Client Credential Configuration when the MEF Client supports device management (either remotely or via manual input). The present document does not specify how the MEF Client Credential Configuration is represented when the MEF Client does not support device management.

8.3.7.2 MEF Client Registration Configuration Details

Purpose: The MEF Client Registration Configuration describes the information provisioned to a MEF Client to enable it to perform MEF procedures authorized by an administrating stakeholder. The administrating stakeholder arranges for the MEF Client Registration Configuration to be provided to the MEF Client.

Pre-conditions:

- The MEF Client and MEF have been configured with credentials which can be used for mutual authentication: see MEF Client Credential Configuration in clause 8.3.7.1.

- If the MEF Client and MEF will use certificates for mutual authentication, then:

  The administrating stakeholder (or another stakeholder acting on behalf of the administrating stakeholder) possesses a copy of the MEF Client's Certificate Information as defined in clause 8.1.2.4. The MEF is provided with a copy of the MEF Client's Certificate Information. The present document does not specify how this information is provided to the MEF by the administrating stakeholder (or another stakeholder acting on behalf of the administrating stakeholder).

  The administrating stakeholder (or another stakeholder acting on behalf of the administrating stakeholder) possesses a copy of the MEF Trust Anchor CA Certificates. The MEF Client is provided with a copy of the MEF Trust Anchor CA Certificates.

- The administrating stakeholder arranges for the MEF to allow the MEF Client to perform MEF Client Registration. This could involve pre-authorization or real-time authorization.

Details:

The MEF Client Registration Configuration (*mefClientRegCfg*) includes the information shown in table 8.3.7.2-1, and has data type sec:clientRegCfg (see clause 12.4.2).

Table 8.3.7.2-1: Information in the MEF Client Registration Configuration

Element Name	Multiplicity	Notes
expirationTime	0..1	Time when the configuration expires
labels	0..1	List of labels to enable discovery of the MEF Client registration record
fqdn	1	MEF-FQDN (also known as MEF-ID)
adminFQDN	1	FQDN of the administrating stakeholder
httpPort	0..1	Port number when using HTTP [i.20]
coapPort	0..1	Port number when using CoAP [i.21]
websocketPort	0..1	Port number when using WebSocket [i.19]

8.3.7.3 MEF Key Registration Configuration Details

Purpose: The MEF Key Registration Configuration describes the information provisioned to a MEF Client to enable it to perform MEF procedures authorized by an administrating stakeholder. The administrating stakeholder arranges for the MEF Client Registration Configuration to be provided to the MEF Client.

Pre-conditions:

- The MEF Client has performed the MEF Client Registration procedure with the MEF for the administrating stakeholder.

- The MEF Client has currently-valid credentials for mutual authentication with the MEF.

Details:

The MEF Key Registration Configuration (*mefKeyRegCfg*) includes the information shown in table 8.3.7.3-1, and has data type sec:keyRegCfg (see clause 12.4.3).

Table 8.3.7.3-1: Information in the MEF Key Registration Configuration

Element Name	Multiplicity	Notes
expirationTime	0..1	Expiration time
labels	0..1	List of labels to enable discovery of the key registration
adminFQDN	1	FQDN of the administrating stakeholder
SUID	1	SUID constraining the usage of the Key Value established during the MEF Key Registration procedure.
targetIDs	0..1	List of identifiers for authorized target MEF Clients

8.3.8 Profile for Device Configuration within an Enrolment Exchange

ETSI TS 118 122 [57] specifies a series of resources types, and procedures on those resource types, for configuration of AEs and CSEs on Field Devices.

As stated in clause 8.3.4.5, Device Configuration can be performed within an Enrolment Exchange with a MEF, or in a DM session with other DM servers (separate from an Enrolment Exchange). When Device Configuration is used with an Enrolment Exchange, then there are two constraints on the Device Configuration <*mgmtObj*> specializations:

[*myCertFileCred*]: This <*mgmtObj*> specialization is not configured by a MEF. Instead, a MEF shall use the Certificate Provisioning procedures of clause 8.3.6 for provisioning a certificate for the MEF Client to use for authenticating itself.

[*authenticationProfile*]: The *symmKeyValue* attribute of this <*mgmtObj*> specialization for provisioning symmetric keys is not used by the MEF (See note below). Instead, a MEF shall use the Symmetric Key Provisioning procedures of clause 8.3.5 for provisioning symmetric keys to the MEF Client. This is achieved in two steps:

1) The MEF uses Device Configuration to configure a MO corresponding to the [*authenticationProfile*], with the following constraints:

   - The [*authenticationProfile*] shall link to the [*MEFClientRegCfg*] associated with the registration on the MEF for the administrating stakeholder which authorized the [*authenticationProfile*].

   - The [*authenticationProfile*] shall include the e*xpirationTime*, and *MAFKeyRegDuration* attributes, and may include the *MAFKeyRegLabels.*

   - The [*authenticationProfile*] resource does not include the values corresponding to the *symmKeyID* and *symmKeyValue* attributes.

2) The MEF subsequently issues a MO_NODE MEF Client Command matching the [*authenticationProfile*] MO node, as described in clause 8.3.4.3. This triggers executing a MEF Key Registration procedure which establishes a symmetric key and symmetric key identifier at the MEF Client and MEF.

NOTE: The *symmKeyValue* is present in the [*authenticationProfile*] for Device Configuration scenarios where the DM Server is not an MEF. The Symmetric Key Provisioning procedures specified in the present document provide greater security, which is why they are mandatory when the DM Server is an MEF.

8.3.9 MEF Client Command Processing

8.3.9.1 Introduction

Purpose: The MEF Client Commands are used by an MEF to control the sequence of Enrolment Exchange procedures executed by the MEF Client.

A MEF Client Command is issued, or reissued, by the MEF to the MEF Client in the response of a MEF Client Command Retrieve procedure (clause 8.3.9.2) or MEF Client Command Update procedure (clause 8.3.9.3). The resulting status, following the attempt to parse and execute the command, is reported to the MEF Client in the request of a MEF Client Command Update procedure (clause 8.3.9.3).

A MEF Client Command Retrieve response or MEF Client Command Update response includes *cmdID*, *cmdDescription*, and initial *cmdStatusCode* of a MEF Client Command being issued, or reissued, by the MEF to the MEF Client. A MEF Client Command Update request, sent from the MEF Client to the MEF, includes *cmdID* and *cmdStatusCode* indicating the result of attempting to parse and execute the command.

These three elements serve the following purposes:

- *cmdID*: disambiguates between sequential MEF Client Commands issued within the context of an MEF Client Registration (which, in some cases, may be one of multiple MEF Client Registrations on the MEF). The *cmdID* serves two purposes: ensuring that a command is not accidentally executed twice; and correlating a MEF Client Command status to the corresponding issued command.

- *cmdDescription*: providing a description of the command to be executed.

- *cmdStatusCode*: enables the MEF to indicate if a command is a reissued command or not, and enables the MEF Client to indicate the result of attempting to parse and execute the command.

MEF Client Command procedures are performed within the context of a (non-expired) MEF Client Registration with the MEF.

8.3.9.2 MEF Client Command Retrieve Procedure

Triggering the Procedure: See clause 8.3.4.6 for mechanisms which can trigger MEF Client Command procedures**.**

Pre-Conditions:

a) The MEF and MEF Client have performed an MEF Handshake, see clause 8.3.5.2.2.

b) The MEF Client Registration is not expired, and the MEF Client has obtained the *MEFClientRegID* for the registration.

c) The MEF Client shall not send an MEF Client Command Retrieve in between the time when an MEF Client receives an MEF Client Command from the MEF and the time when the MEF Client sends a corresponding MEF Client Command Update to report on the status.

d) The MEF Client shall not initiate an MEF Client Command Retrieve while waiting for a MEF Client Command Response from the MEF, unless the MEF takes too long to respond. The time duration for waiting for a response from the MEF is an implementation-specific decision of the MEF Client.

Procedure:

1) The MEF Client shall send a MEF Client Command Retrieve request including the information shown in table 8.3.9.2-1.

Table 8.3.9.2-1: MEF Client Command Retrieve Request message information

Element	Description	Multiplicity
MEFClientRegID	Identifier for the MEF Client registration record for which the MEF Client Command is being requested. See Pre-Condition B. This is the resource identifier of the parent <*mefClientReg*> resource of the MEF Client Command.	1

2) Upon receiving the request, the MEF shall process the request. If error cases are encountered, then the MEF shall send an error response.

3) If the request is processed successfully, then the MEF shall attempt to retrieve the MEF Client Command currently associated with the identified MEF Client registration record:

- If there are currently no more MEF Client Commands to be issued to the MEF Client, then the MEF forms the *cmdDescription* as specified in clause 8.3.9.6.
- If MEF Client Command will trigger a Certificate Provisioning procedure, then the MEF forms the *cmdDescription* as specified in clause 8.3.9.7.
- If MEF Client Command will trigger Device Configuration, then the MEF forms the *cmdDescription* as specified in clause 8.3.9.8.
- In the case of an MO_NODE MEF Client Command, then the MEF forms the *cmdDescription*, as specified in clause 8.3.9.9.

4) The MEF shall compose a MEF Client Command Retrieve response containing the following parameters.

Table 8.3.9.2-2: MEF Client Command Retrieve Response message information

Element	Description	Multiplicity
cmdID	An identifier for a MEF Client Command issued by the MEF (see data type definition in clause 8.6.1 of ETSI TS 118 132 [58])	1
cmdDescription	Description of the MEF Client Command being issued or reissued.	1
cmdStatusCode	*cmdStatusCode* set to MEF_CLIENT_CMD_ISSUED or MEF_CLIENT_CMD_REISSUED as appropriate (see clauses 8.3.9.5.2 and 8.3.9.5.3)	1

The MEF shall send the response to the MEF Client.

5) The MEF Client shall attempt to parse and execute the response information

   a) The MEF Client attempts to parse the Response message information into *cmdID*, *cmdDescription* and *cmdStatusCode* elements; and parse *cmdDescription* into its constituent *cmdClassID* and *cmdArgs* elements. If parsing succeeds, then the MEF client proceeds to step 5b. If parsing fails, then the MEF Client may choose to exit the procedure, or may return to step 1.

   b) The MEF Client compares the *cmdID* of the Response message with *cmdID* sent in the most recent MEF Client Command Update procedure. If the *cmdID* values are distinct, then the MEF Client proceeds to step 5.c. If the *cmdID* values are identical then the MEF_Client exits the procedure, triggering MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_REPEATED_CMD_ID.

   c) The MEF Client interprets *cmdClassID*, to determine the corresponding MEF Client Command Class. If the MEF Client supports the MEF Client Command Class then the MEF Client proceeds to step 5.d. If the MEF Client does not support the MEF Client Command Class then the MEF Client exits the procedure, triggering MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_CLASS_NOT_SUPPORTED.

   d) The MEF Client initiates the MEF Client Command Class-specific procedures:

      - The NO_MORE_COMMANDS MEF Client Command Class-specific procedures are specified in clause 8.3.9.6.
      - The CERT_PROV MEF Client Command Class-specific procedures are specified in clause 8.3.9.7.
      - The DEV_CFG MEF Client Command Class-specific procedures are specified in clause 8.3.9.8.
      - The MO_NODE MEF Client Command Class-specific procedures are specified in clause 8.3.9.9.

8.3.9.3 MEF Client Command Update procedure

Triggering: The MEF Client shall initiate the MEF Client Command Update procedure only when triggered from within the MEF Client Command Retrieve procedure (clause 8.3.9.2), or MEF Client Command Update procedure (as defined in the present clause) or a MEF Client Command Class-specific procedure (clauses 8.3.9.6, 8.3.9.7, 8.3.9.8 and 8.3.9.9). The trigger includes the values for *cmdID* and *cmdStatusCode*.

Pre-Conditions:

A. The MEF and MEF Client have performed an MEF Handshake, see clause 8.3.5.2.2.

B. The MEF Client Registration is not expired, and the MEF Client has obtained the *MEFClientRegID* for the registration.

Procedure:

1) The MEF Client shall send a MEF Client Command Update request including the information shown in table 8.3.9.3-1.

Table 8.3.9.3-1: MEF Client Command Update Request message information

Element	Description	Multiplicity
MEFClientRegID	Identifier for the MEF Client registration record for which the MEF Client Command is being requested. See Pre-Condition B.	1
cmdID	Provided when the procedure was triggered	1
cmdStatusCode	Provided when the procedure was triggered	1

2) Upon receiving the request, the MEF shall process the request.

   a) The MEF attempts to parse the request message information into *MEFClientRegID, cmdID* and *cmdStatusCode* elements. If parsing succeeds, then the MEF proceeds to step 2b. If parsing fails, then the MEF shall send an MEF Client Command Update response with the BAD_REQUEST request Status Code from table 5.1.2-3 in ETSI TS 118 132 [58], and no further steps are performed.

   b) The MEF compares the *cmdID* in the request to the *cmdID* of the most recently issued MEF Client Commands. If there is no match then the *cmdID* and *cmdStatusCode* are discarded. If there is a match, then the MEF can records the *cmdID* and *cmdStatusCode.*

3) The MEF determines the next MEF Client Command to issue to the MEF Client, as described in step 3 of clause 8.3.9.2.

4) The MEF shall compose a MEF Client Command Update response containing the same elements as an MEF Client Command Retrieve response, shown in table 8.3.9.2-2 in step 4 of clause 8.3.9.2. The MEF shall send the response to the MEF Client.

5) The MEF Client shall attempt to parse and execute the response message information, as specified in step 5 of clause 8.3.9.2.

8.3.9.4 The cmdDescription element

The *cmdDescription* element has data type sec:cmdDescription defined in clause 12.4.4 and includes the following elements:

- *cmdClassID:* identifying a class of MEF Client Commands.
- (conditional on *cmdClassID*) *cmdArgs:* containing arguments specific to the *cmdClass.*
- (optional) *targetID*: When the MEF Client is a Node acting on behalf of a CSE and/or multiple AE, then the *targetID* identifies to which entity the command applies.

MEF Client Command processing supports the following classes of MEF Client Commands:

- NO_MORE_COMMANDS: indicating that the MEF has no more MEF Client Commands for the MEF Client. The NO_MORE_COMMANDS MEF Client Command is specified in clause 8.3.9.6.
- CERT_PROV: for triggering Certificate Provisioning procedures. The CERT_PROV MEF Client Commands are specified in clause 8.3.9.7.
- DEV_CFG: for triggering Device Configuration. The DEV_CFG MEF Client Commands are specified in clause 8.3.9.8.
- MO_NODE: for triggering procedures. The CERT_PROV MEF Client Commands are specified in clause 8.3.9.9.

8.3.9.5 The *cmdStatusCode* element

8.3.9.5.1 Introduction

The *cmdStatusCode* is used by the MEF and MEF Client to indicate the status of an issued Command. The value in the *cmdStatusCode* element is of datatype sec:cmdStatusCode, specified in clause 12.3.2.4. Table 8.3.9.5.1-1 provides an informative summary of the *cmdStatusCode*, with normative description in the remaining clauses of 8.3.9.5.

Table 8.3.9.5.1-1: Overview of the *cmdStatusCode* element

cmdStatusCode	Assigned by	*cmdClass* of issued MEF Client Command	Clause
MEF_CLIENT_CMD_ISSUED	MEF	Any	8.3.9.5.2
MEF_CLIENT_CMD_REISSUED	MEF	Any	8.3.9.5.3
MEF_CLIENT_CMD_OK	MEF Client	Any. See note.	8.3.9.5.4
MEF_CLIENT_CMD_REPEATED_CMD_ID	MEF Client	Any	8.3.9.5.5
MEF_CLIENT_CMD_CLASS_NOT_SUPPORTED	MEF Client	Any. See note.	8.3.9.5.6
MEF_CLIENT_CMD_BAD_ARGUMENTS	MEF Client	Any.	8.3.9.5.7
MEF_CLIENT_CMD_UNACCEPTABLE_ARGUMENTS	MEF Client	CERT_PROV, DEV_CFG, MO_NODE	8.3.9.5.8
MEF_CLIENT_CMD_CERT_PROV_SERVER_ERROR	MEF Client	CERT_PROV	8.3.9.5.9
MEF_CLIENT_CMD_CERT_PROV_CLIENT_ERROR	MEF Client	CERT_PROV	8.3.9.5.10
MEF_CLIENT_CMD_DEV_CFG_SERVER_ERROR	MEF Client	DEV_CFG	8.3.9.5.11
MEF_CLIENT_CMD_DEV_CFG_CLIENT_ERROR	MEF Client	DEV_CFG	8.3.9.5.12
MEF_CLIENT_CMD_MO_NODE_NOT_FOUND	MEF Client	MO_NODE	8.3.9.5.13
MEF_CLIENT_CMD_MO_NODE_TYPE_CONFLICT	MEF Client	MO_NODE	8.3.9.5.14
MEF_CLIENT_CMD_MO_NODE_BAD_ARGS	MEF Client	MO_NODE	8.3.9.5.15
MEF_CLIENT_CMD_MO_NODE_UNACCEPTABLE_ARGS	MEF Client	MO_NODE	8.3.9.5.16
MEF_CLIENT_CMD_MO_NODE_INCONSITENT_CONFIG	MEF Client	MO_NODE	8.3.9.5.17
MEF_CLIENT_CMD_MO_NODE_PROCESSING_FAILED	MEF Client	MO_NODE	8.3.9.5.18
NOTE: In normal circumstances, an MEF Client should not provide this *cmdStatusCode* when the issued command has *cmdClass* indicating NO_MORE_COMMANDS.			

8.3.9.5.2 *cmdStatusCode* MEF_CLIENT_CMD_ISSUED

The MEF is issuing the command, and the MEF expects that this is the first time that the MEF Client is receiving the command.

8.3.9.5.3 *cmdStatusCode* MEF_CLIENT_CMD_REISSUED

The MEF previously issued the command, and did not receive a corresponding MEF Client Command Update from the MEF Client (in particular, providing the status of the executed command), and consequently the MEF is reissuing the command.

If the MEF Client already performed the command, then it reports the status, otherwise the MEF Client performs the command.

8.3.9.5.4 *cmdStatusCode* MEF_CLIENT_CMD_OK

The MEF Client successfully performed the command.

8.3.9.5.5 *cmdStatusCode* MEF_CLIENT_CMD_REPEATED_CMD_ID

The *cmdID* in the MEF Client Command Response message matches the *cmdID* sent in the most recent MEF Client Command Update procedure. This would indicate some processing error on the MEF.

8.3.9.5.6 *cmdStatusCode* MEF_CLIENT_CMD_CLASS_NOT_SUPPORTED

The MEF Client does not support the requested *cmdClass*.

8.3.9.5.7 *cmdStatusCode* MEF_CLIENT_CMD_BAD_ARGUMENTS

The MEF Client supports the *cmdClass*, but the MEF Client could not parse *cmdArgs*.

8.3.9.5.8 *cmdStatusCode* MEF_CLIENT_CMD_UNACCEPTABLE_ARGUMENTS

The MEF Client supports the *cmdClass*, and the MEF Client parsed *cmdArgs* element, but at least one of the elements of *cmdArgs* which could be processed by the MEF Client had an unacceptable value.

8.3.9.5.9 *cmdStatusCode* MEF_CLIENT_CMD_CERT_PROV_SERVER_ERROR

The MEF Client supports the CERT_PROV *cmdClass*, and the MEF Client parsed the *cmdArgs* element, but was unable to perform the command due to some error regarding communicating with the Certificate Provisioning server or an error internal to the Certificate Provisioning server.

8.3.9.5.10 *cmdStatusCode* MEF_CLIENT_CMD_CERT_PROV_CLIENT_ERROR

The MEF Client supports the CERT_PROV *cmdClass*, and the MEF Client parsed the *cmdArgs* element, but was unable to perform the command due to a processing error on the MEF Client.

8.3.9.5.11 *cmdStatusCode* MEF_CLIENT_CMD_DEV_CFG_SERVER_ERROR

The MEF Client supports the DEV_CFG *cmdClass*, and the MEF Client parsed the *cmdArgs* element, but was unable to perform the command due to some error regarding communicating with the Device Configuration server or an error internal to the Device Configuration server.

8.3.9.5.12 *cmdStatusCode* MEF_CLIENT_CMD_DEV_CFG_CLIENT_ERROR

The MEF Client supports the DEV_CFG *cmdClass*, and the MEF Client parsed the *cmdArgs* element, but was unable to perform the command due to some issue a processing error on the MEF Client.

8.3.9.5.13 *cmdStatusCode* MEF_CLIENT_CMD_MO_NODE_NOT_FOUND

The MEF Client supports the MO_Node *cmdClass*, and parsed the *cmdArgs* element, but could not find the MO node at the *objectPath* element in the MEF Client Command arguments.

8.3.9.5.14 *cmdStatusCode* MEF_CLIENT_CMD_MO_NODE_TYPE_CONFLICT

The MEF Client supports the MO_Node cmdClass, parsed the *cmdArgs* element, and found the MO node at the *objectPath* element, but the type of the MO Node does not match the *objectType* element in the MEF Client Command arguments.

8.3.9.5.15 *cmdStatusCode* MEF_CLIENT_CMD_MO_NODE_BAD_ARGS

The MEF Client supports the MO_Node *cmdClass*, parsed the *cmdArgs* element, and found the MO node at the *objectPath* element, and the type of the MO Node matches the *objectTypeID* element, but the MEF Client could not parse the *objectTypeSpecifcArgs elememt.*

8.3.9.5.16 *cmdStatusCode* MEF_CLIENT_CMD_MO_NODE_UNACCEPTABLE_ARGS

The MEF Client supports the MO_Node *cmdClass*, parsed the *cmdArgs* element, and found the MO node at the *objectPath* element, and the type of the MO Node matches the *objectType* element, and the MEF Client could not parse the *objectTypeSpecifcArgs* element but the remaining MO-type-specific arguments in MEF Client Command arguments are unacceptable to the MEF Client.

8.3.9.5.17 *cmdStatusCode* MEF_CLIENT_CMD_MO_NODE_INCONSISTENT_CONFIG

The MEF Client supports the MO_Node *cmdClass*, parsed the *cmdArgs* element, and found the MO node at the *objectPath* element, and the type of the MO Node matches the *objectTypeID* element, and the remaining MO-type-specific arguments in MEF Client Command arguments are acceptable, but the configuration of the MO nodes is inconsistent and preventing MO Node processing.

8.3.9.5.18 *cmdStatusCode* MEF_CLIENT_CMD_MO_NODE_PROCESSING_FAILED

The MEF Client supports the MO_Node *cmdClass*, parsed the *cmdArgs* element, and found the MO node at the *objectPath* element, and the type of the MO Node matches the *objectType* element, and the remaining MO-type-specific arguments in MEF Client Command arguments are acceptable, but there has been some other error in executing the MO Node processing.

8.3.9.6 NO_MORE_COMMANDS MEF Client Command Class-specific Processes

Purpose: When *cmdClassID* indicates NO_MORE_COMMANDS, then the MEF is indicating that it has no more MEF Client Commands for the MEF Client.

Elements of *cmdArgs*: If *cmdDescription* contains *cmdClassID* indicating NO_MORE_COMMANDS, then *cmdArgs* shall contain the *noMoreCmdArgs* element of data type sec:noMoreCmdArgs which includes the following elements:

- *retryDuration*: indicating a time duration, after which the MEF Client is expected to attempt MEF Client Command Retrieve. The *retryDuration* is cancelled if the MEF Client successfully performs another MEF Client Command procedure, within the scope of the MEF Client Registration, before the *retryDuration* completes. See clause 8.3.4.6 for other mechanisms which can trigger MEF Client Command procedures.

Forming *cmdDescription*:

1) The MEF shall form *cmdArgs* containing the elements described in "Elements of cmdArgs" above:

   - *retryDuration*: set to the time duration before the MEF wishes the MEF Client to attempt the next MEF Client Command Retrieve, with the understanding that the *retryDuration* is cancelled if the MEF Client successfully performs another MEF Client Command procedure.

2) The MEF shall form *cmdDescription* with *cmdClassID* indicating NO_MORE_COMMANDS and *cmdArgs* formed in step 1.

Parsing and Executing *cmdArgs*:

1) The MEF Client shall attempt to parse *cmdArgs* into the elements described in "Elements of cmdArgs". If the parsing succeeds, then the MEF Client proceeds to step 4. If parsing fails, then the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_BAD_ARGUMENTS.

2) The MEF Client MEF Client Command will not perform a MEF Client Command Retrieve procedure or MEF Client Command Update procedure unless triggered.

3) The MEF Client shall set a timer based on *retryDuration*:

   a) The timer shall be cancelled if, by some other mechanism as described in clause 8.3.4.6, the MEF Client is triggered to performs an MEF Client Command Retrieve procedure or MEF Client Command Update procedure before the timer expires.

   b) If the time expires, then the MEF Client shall perform an MEF Client Command Retrieve procedure, (clause 8.3.9.2), at some time selected by the MEF Client.

8.3.9.7 CERT_PROV MEF Client Command Class-specific Processes

Purpose: When *cmdClassID* indicates CERT_PROV, then the MEF is indicating that the MEF Client is to perform a Certificate Provisioning Procedure with the MEF.

Elements of *cmdArgs*: If *cmdDescription* contains *cmdClassID* indicating CERT_PROV, then the *cmdArgs* shall contain the *certProvArgs* element of data type sec:certProvArgs which includes the following elements:

- *certProvProtocol*: indicating the Certificate Provisioning protocol (EST or SCEP) that the MEF Client is to use.
- *URI*: indicating the base URI to be used for the indicated Certificate Provisioning protocol.
- *certSubjectType*: indicating if the subject of the provisioned certificate will be a Node, CSE or AE.
- *certSubjectID*: the Node-ID or CSE-ID or AE-ID of the subject of the certificate

Forming *cmdDescription*:

1) The MEF shall form *cmdArgs* containing the elements described in "Elements of *cmdArgs*" above:
   - *certProvProtocol*: The MEF shall assign this element to indicate the protocol (EST or SCEP) that the MEF Client is to use for Certificate Provisioning.
   - *URI*: The MEF shall assign this element to base URI to be used for the indicated Certificate Provisioning protocol. The FQDN of the base URI shall match the FQDN of the MEF issuing the MEF Client Command.
   - *certSubjectType*: The MEF Client shall assign this element to indicate if the subject of the provisioned certificate will be a Node, CSE or AE.
   - *certSubjectID:* the MEF shall assign this element to be the Node-ID or CSE-ID or AE-ID of the subject of the certificate.
2) The MEF shall form *cmdDescription* with *cmdClassID* indicating CERT_PROV and *cmdArgs* formed in step 1.

Parsing and Executing *cmdArgs*:

1) See step 3 in clause 8.3.9.6.
2) The MEF Client shall verify that the *cmdArgs* elements are acceptable:
   - *certProvProtocol*: Verification of this element succeeds only if the indicated protocol (EST or SCEP) is supported by the MEF Client. If verification succeeds, then the MEF Client selects the Certificate Provisioning protocol indicated by the element.
   - *URI:* Verification of this element succeeds only if the FQDN of the base URI matches the FQDN of the MEF issuing the MEF Client Command. If verification succeeds, then the MEF Client set the base URI to the value in this element.
   - *certSubjectType*:
     - If the MEF Client is in a Node acting on behalf of a CSE and/or multiple AE then verification of this element succeeds only if the value indicates a Node, CSE or AE.
     - If the MEF Client is in a CSE, then verification of this element succeeds only if the value indicates a CSE.
     - If the MEF Client is in a CSE, then verification of this element succeeds only if the value indicates a CSE.

       If verification succeeds, then the MEF Client records the *certSubjectType* as the certificate subject type.
   - *certSubjectID*: Verification of this element depends on the value of *certSubjectType:*
     - If *subjectType* indicates a Node, then verification of this element succeeds only if the value is a Node-ID.
     - If *certSubjectType* indicates a CSE, then verification of this element succeeds only if the value is a AE-ID.

- If *certSubjectID* indicates a AE, then verification of this element succeeds only if the value is a CSE-ID.

  If verification succeeds, then the MEF Client records the *certSubjectID* as the certificate subject identity.

If the verification of any argument fails, then the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_UNACCEPTABLE_ARGUMENTS.

3) The MEF Client shall attempt executing the selected Certificate Provisioning procedure (EST specified in clause 8.3.6.2 or SCEP specified in clause 8.3.6.3), with base URI, certificate subject type and certificate subject identity as determined in step 4. The certificate subject identity shall be provided in the SubjectAltName extension of the certificate signing request.

4) Following the attempt to execute the selected Certificate Provisioning procedure, the MEF Client shall perform the MEF Client Command Update Procedure with *cmdID* assigned to the *cmdID* of the received command and *cmdStatusCode* assigned as follows:

   - If the Certificate Provisioning procedure completed successfully, then the MEF Client shall set *cmdStatusCode* to the value for MEF_CLIENT_CMD_OK.

   - If the Certificate Provisioning procedure did not complete successfully due to an error regarding communicating with the Certificate Provisioning server or an error internal to the Certificate Provisioning server, then the MEF Client shall set *cmdStatusCode* to the value for MEF_CLIENT_CMD_CERT_PROV_SERVER_ERROR.

   - If the Certificate Provisioning procedure did not complete successfully due to an error occurring in the Certificate Provisioning client (in the MEF Client), then the MEF Client shall set *cmdStatusCode* to the value for MEF_CLIENT_CMD_CERT_PROV_CLIENT_ERROR.

8.3.9.8 DEV_CFG MEF Client Command Class-specific Processes

Purpose: When *cmdClassID* indicates DEV_CFG, then the MEF is indicating that the MEF Client is to perform a Device Configuration (ETSI TS 118 122 [57]) with the MEF Client acting as the DM Client and the MEF acting as the DM Server.

Elements of *cmdArgs*: If *cmdDescription* contains *cmdClassID* indicating DEV_CFG, then the *cmdArgs* shall contain the *devCfgArgs* element which includes the following elements:

- *devMgmtID*: indicating the DM protocol (e.g. OMA DMv1.3, OMA DMv2.0, OMA LwM2M, BBF TR-069) that the MEF Client is to use for Device Configuration.
- *URI*: URI of the DM Server.

Forming *cmdDescription*:

1) The MEF shall form *cmdArgs* containing the elements described in "Elements of cmdArgs" above:

   - *devMgmtID*: The MEF shall assign this element to indicate the protocol (e.g. OMA DMv1.3, OMA DMv2.0, OMA LwM2M, BBF TR-069 [i.31]) that the MEF Client is to use for Device Configuration.

   - *URI*: The MEF shall assign this element to the URI of the DM Server. The FQDN of the base URI shall match the FQDN of the MEF issuing the MEF Client Command.

2) The MEF shall form *cmdDescription* with *cmdClassID* indicating DEV_CFG and *cmdArgs* formed in step 1.

Parsing and Executing *cmdArgs*:

3) See step 3 in clause 8.3.9.6.

4) The MEF Client shall verify that the *cmdArgs* elements are acceptable:

   - *devMgmtID*: Verification of this element succeeds only if the indicated protocol (e.g. OMA DMv1.3, OMA DMv2.0, OMA LwM2M, BBF TR-069) is supported by the MEF Client. If verification succeeds, then the MEF Client selects the DM protocol indicated by the element.

   - *URI:* Verification of this element succeeds only if the FQDN of the base URI matches the FQDN of the MEF issuing the MEF Client Command. If verification succeeds, then the MEF Client set the DM Server URI to the value in this element.

     If the verification of any argument fails, then the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_UNACCEPTABLE_ARGUMENTS.

5) The MEF Client shall attempt executing Device Configuration per ETSI TS 118 122 [57] using the DM protocol and DM Server URI determined in step 4.

6) Following the attempt to execute Device Configuration, the MEF Client shall perform the MEF Client Command Update Procedure with *cmdID* assigned to the *cmdID* of the received command and *cmdStatusCode* assigned as follows:

   - If the Device Configuration procedure completed successfully, then the MEF Client shall set *cmdStatusCode* to the value for MEF_CLIENT_CMD_OK.

   - If the Device Configuration procedure did not complete successfully due to an error regarding communicating with the DM server or an error internal to the DM server, then the MEF Client shall set *cmdStatusCode* to the value for MEF_CLIENT_CMD_DEV_CFG_SERVER_ERROR.

   - If the Device Configuration procedure did not complete successfully due to an error occurring in the DM client (in the MEF Client), then the MEF Client shall set *cmdStatusCode* to the value for MEF_CLIENT_CMD_DEV_CFG_CLIENT_ERROR.

8.3.9.9 MO_NODE MEF Client Command Class-specific Processes

8.3.9.9.1 Generic MO_NODE Processes

Purpose: When *cmdClassID* indicates MO_NODE, then the MEF is indicating that the MEF Client is to process an MO_NODE that has been already configured to the DM Client of the MEF Client (e.g. using Device Configuration in ETSI TS 118 122 [57]).

Elements of *cmdArgs*: If *cmdDescription* contains *cmdClassID* indicating MO_NODE, then the *cmdArgs* shall contain the *MONodeCmdArgs* element which includes the following elements:

- *objectPath*: the path of the MO node to be processed.
- *objectTypeID*: indicating the type of the specialization of the <*mgmtObj*> resource which provides the data model for the MO node to be processed.
- *(Optional) objectTypeSpecificArgs*: additional arguments dependent on type of the specialization of the <*mgmtObj*> resource (see *objectTypeID*):

  - If *objectTypeID* matches the [*authenticationProfile*] specialization of the <*mgmtObj*>resource, then *objectTypeSpecificArgs* is present, and is defined in clause 8.3.9.9.2.

  - For all other specializations, this element is not present.

Forming *cmdDescription*:

1) The MEF shall form *cmdArgs* containing the elements described in "Elements of *cmdArgs*" above:

   - *objectPath*: The MEF shall assign this element to the path of the MO node to be processed.

   - *objectTypeID*: The MEF shall assign this element to the identifier of the type of MO node to be processed.

- *(Optional) objectTypeSpecificArgs*:

  - If *objectTypeID* matches the [*authenticationProfile*] specialization of the <*mgmtObj*>resource, then *objectTypeSpecificArgs* is formed as specified in "Forming *objectTypeSpecificArgs*" in clause 8.3.9.9.2.

  - For all other specializations, this element is not present.

2) The MEF shall form *cmdDescription* with *cmdClassID* indicating MO_NODE and *cmdArgs* formed in step 1.

Parsing and Executing *cmdArgs*:

3) See step 3 in clause 8.3.9.6.

4) The MEF Client shall verify that the *cmdArgs* elements are acceptable:

   a) *objectPath:* Verification of these elements succeeds only if there is an MO node addressed by the *objectPath*. If verification succeeds the MEF Client proceeds to step 4b. If the verification of this argument fails, then the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_MO_NODE_NOT_FOUND.

   b) *objectTypeID:* Verification of these elements succeeds only if *objectTypeID* matches the type MO node addressed by the *objectPath* (see step 4a). If verification succeeds the MEF Client proceeds to step 5. If the verification of this argument fails, then the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_MO_NODE_TYPE_CONFLICT.

5) The MEF Client applies the processing specific to the *objectTypeID*:

- If *objectTypeID* matches the [*authenticationProfile*] specialization of the <*mgmtObj*> resource, then the MEF Client shall perform "Processing an [*authenticationProfile*] MO Node" in clause 8.3.9.9.2.

- If *objectTypeID* matches the [*trustAnchorCred*] specialization of the <*mgmtObj*> resource, then the MEF Client shall perform "Processing a [*trustAnchorCred*] MO Node" in clause 8.3.9.9.7.

- If *objectTypeID* matches the [*MAFClientCfgReg*] specialization of the <*mgmtObj*> resource, then the MEF Client perform "Processing a [*MAFClientCfgReg*] MO Node" in clause 8.3.9.9.8.

8.3.9.9.2 [*authenticationProfile*]-specific Processes

Purpose: Processing an [*authenticationProfile*] MO node ensures that the MEF Client has been able to establish the credentials needed to use that [*authenticationProfile*] MO node for mutual authentication.

Elements of *objectTypeSpecificArgs*: when objectTypeID matches the [*authenticationProfile*] specialization of the <*mgmtObj*> resource, then the *objectTypeSpecificArgs* element shall be present and shall contain the *authProfileMONodeArgs* element which includes the following elements:

- *SUID:* this value matches the SUID in the addressed MO Node.

Forming *objectTypeSpecificArgs*:

1) The MEF shall form *objectTypeSpecificArgs* containing *authProfileMONodeArgs* with the elements described in "Elements of *objectTypeSpecificArgs*" above:

   - *SUID*: The MEF shall assign this element to the value in the *SUID* element expected to be in the MO node located at the *objectPath* on the MEF Client.

Processing an [*authenticationProfile*] MO Node:

2) The MEF Client shall attempt to parse *objectTypeSpecificArgs* into the elements described in "Elements of *objectTypeSpecificArgs*". If the parsing succeeds, then the MEF Client proceeds to step 3. If parsing fails, then the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_MN_NODE_BAD_ARGS.

3) The MEF Client shall verify that the *objectTypeSpecificArgs* elements are acceptable:

   - *SUID:* Verification succeeds if the SUID in *objectTypeSpecificArgs* matches the *SUID* in the addressed MO_NODE.

   If *objectTypeSpecificArgs* elements verification succeeds, then the MEF Client proceeds to step 5. If *objectTypeSpecificArgs* elements verification fails for any element, then the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_MO_NODE_UNACCEPTABLE_ARGS.

4) The MEF Client shall verify that the *SUID* matches the configuration of the [*authenticationProfile*] MO Node and its parent and child MO Nodes:

   - If the *SUID* is in the set {11, 21, 31, 41}, then verification shall fail if the parent MO Node of the [*authenticationProfile*] MO Node does not correspond to the [*MAFClientRegCfg*] specialization of the <*mgmtObj*> resource,

   - If the *SUID* is in the set {12, 22, 32, 42}, then verification shall fail if the parent MO Node of the [*authenticationProfile*] MO Node does not correspond to the [*registration*] specialization of the <*mgmtObj*> resource.

   - If the *SUID* is in the set {13, 23, 33, 43}, then verification shall fail if the parent MO Node of the [*authenticationProfile*] MO Node does not correspond to the [*dataCollection*] specialization of the <*mgmtObj*> resource.

   - If the *SUID* is in the set {11,12,13}, then verification shall fail if the *symmKeyID* attribute is not present in the [*authenticationProfile*] MO Node.

   - If the *SUID* is in the set {21,22,23}, then verification shall fail if:

     - The *keyRegDuration* attribute is not present in the [*authenticationProfile*] MO Node, or

     - The child MO Node of the [*authenticationProfile*] MO Node does not correspond to the [*MEFClientRegCfg*] specialization of the <*mgmtObj*> resource.

   - If the *SUID* is in the set {31,32,33}, then verification shall fail if:

     - The *keyRegDuration* attribute is not present in the [*authenticationProfile*] MO Node, or

     - The child MO Node of the [*authenticationProfile*] MO Node does not correspond to the [*MAFClientRegCfg*] specialization of the <*mgmtObj*> resource.

   - If the *SUID* is in the set {11, 12, 21, 22, 31, 32}, then verification shall fail if:

     - The *TLSCiphersuites* attribute is not present in the [*authenticationProfile*] MO Node, or

     - The *TLSCiphersuites* attribute is present but does not include the mandatory DTLS or TLS Ciphersuites for TLS-PSK-Based Security Frameworks in clause 10.2.2.

   - If the *SUID* is in the set {41, 42, 43}, then verification shall fail if:

     - The *myCertFingerprint* attribute is not present in the [*authenticationProfile*] MO Node, or

     - The *TLSCiphersuites* attribute is not present in the [*authenticationProfile*] MO Node or the *TLSCiphersuites* attribute is present but does not include the mandatory DTLS or TLS Ciphersuites for Certificate-Based Security Frameworks in clause 10.2.3, or

     - The [*authenticationProfile*] MO Node has one or more child MO nodes corresponding to the [*trustAnchorCred*] specialization of the <*mgmtObj*> resource.

   If verification succeeds, then the MEF Client proceeds to step 5. If the verification of this argument fails for any element, then the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_MO_NODE_INCONSISTENT_CONFIG.

5) The MEF Client applies the processing specific to the *SUID*:

- If the *SUID* is in the set {11,12,13}, corresponding to a pre-provisioned symmetric key SUID, then the MEF Client performs "Process [authenticationProfile] MO Node with pre-provisioned symmetric key SUID" in clause 8.3.9.9.3.
- If the *SUID* is in the set {21,22,23}, corresponding to a MEF-established symmetric key SUID, then the MEF Client performs "Process [authenticationProfile] MO Node with MEF-established symmetric key SUID" in clause 8.3.9.9.4.
- If the *SUID* is in the set {31,32,33}, corresponding to a MAF-established symmetric key SUID, then the MEF Client performs "Process [authenticationProfile] MO Node with MAF-established symmetric key SUID" in clause 8.3.9.9.5.
- If the *SUID* is in the set {41,42,43}, corresponding to a certificate SUID, then the MEF Client performs "Process [authenticationProfile] MO Node with certificate SUID" in clause 8.3.9.9.6.

8.3.9.9.3 Process [*authenticationProfile*] MO Node with pre-provisioned symmetric key

Purpose: Processing an [*authenticationProfile*] MO node with pre-provisioned symmetric key SUID (in the set {11,12,13}) ensures that the MEF Client has access to a local copy of the pre-provisioned symmetric key, for subsequent use with the [*authenticationProfile*] MO node.

Preconditions:

A. This procedure will succeed only if there is a local copy of the pre-provisioned symmetric key value which can be accessed by the MEF.

Procedure:

1. The MEF Client shall determine if the *symmKeyValue* attribute is present in the [*authenticationProfile*] MO node:
   - If the attribute is not present, then the MEF Client shall proceed to step 2.
   - If the attribute is present, then the MEF Client has access to the pre-provisioned symmetric key value. The MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_OK.
2. The MEF Client obtains the value of the *symmKeyID* attribute of the [*authenticationProfile*] MO node. The MEF Client determines if the MEF Client has a local copy of the symmetric key value with identifier matching the *symmKeyID* attribute:
   - If a local copy of the symmetric key value is not present, then the MEF Client does not have access to the pre-provisioned symmetric key value. The MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_MO_NODE_PROCESSING_FAILED.
   - If a local copy of the symmetric key value is present, then the MEF Client has access to the pre-provisioned symmetric key value. The MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_OK.

8.3.9.9.4 Process [*authenticationProfile*] MO Node with MEF-established symmetric key

Purpose: Processing an [*authenticationProfile*] MO node with MEF-established symmetric key SUID (in the set {21, 22, 23}) ensures that the MEF Client establishes a symmetric key with the MEF for subsequent use with the [*authenticationProfile*] MO node.

Preconditions:

A. This procedure will succeed only if the [*authenticationProfile*] MO node has a child [*MEFClientRegCfg*] MO node and a parent MO Node which may be of type [*MAFClientRegCfg*], [*registration*] or [*dataCollection*].

B. This procedure assumes that the MEF Client has a currently-valid MEF Client Registration with the MEF and administrating stakeholder identified in the child [*MEFClientRegCfg*] MO node.

Procedure: The MEF Client shall attempt the MEF Key Registration procedure as described in clause 8.3.5.2.7, with the MEF Client acting as the Source MEF Client and with following clarifications:

At step 4 in clause 8.3.5.2.7, the MEF Client shall form the MEF Key Registration Request (see table 8.3.5.2.7-1) as follows:

- *MEFFQDN*: shall be set to the value of the *fqdn* attribute in the child [*MEFClientRegCfg*] MO node;
- *expirationTime* shall be computed by adding the current time to the value of the *keyRegDuration* in the [*authenticationProfile*] MO node;
- *labels* shall be assigned the value of the *keyRegLabels* attribute in the [*authenticationProfile*] MO node;
- *adminFQDN*: shall be assigned to the value of the *adminFQDN* attribute in the child [*MEFClientRegCfg*] MO node;
- *SUID*: shall be assigned to the value of the *SUID* attribute in the child [*authenticationProfile*] MO node;
- *targetIDs*: shall be assigned according to the parent MO node of the [*authenticationProfile*] MO node:
  - In the case of a parent [*registration*] MO node, the *targetIDs* shall be assigned the CSE-ID of the Registrar CSE.
  - In the case of a parent [*dataCollection*] MO node, the *targetIDs* shall be assigned the CSE-ID determined from the *containerPath* attribute of the [*dataCollection*] MO node.
  - In the case of a parent [*MAFClientRegCfg*] MO node, the *targetIDs* shall be assigned the *fqdn* attribute of the [*MAFClientRegCfg*] MO node.
- *keyValue*: shall not be present.

The MEF Client shall select a protocol (HTTP, CoAP, WebSocket) based on the protocol(s) supported by the MEF Client, and the protocol(s) supported by the MEF as indicated by the presence of the *httpPort*, *coapPort* and *websocketPort* in the child [*MEFClientRegCfg*] MO node. The MEF Client shall use the port number provided in the appropriate *httpPort*, *coapPort* or *websocketPort* attribute of the child [*MEFClientRegCfg*] MO node.

At step 10 in clause 8.3.5.2.7, if the MEF Client receives a successful MEF Key Registration Response (see table 8.3.5.2.7-2) then the MEF Client shall verify the following information received in the MEF Key Registration Response:

- *expirationTime:* Verification fails if this value is prior to the time when verification is performed.
- *Source MEF Client ID:* Verification fails if this value is distinct from the identifier of the MEF Client.
- *adminFQDN*: Verification fails if this value is distinct from the corresponding value sent in the MEF Key Registration Request in step 4 of clause 8.3.5.2.7.
- *SUID*: Verification fails if this value is distinct from the corresponding value sent in the MEF Key Registration Request in step 4 of clause 8.3.5.2.7.
- *targetIDs*: Verification fails if this value is distinct from the corresponding value sent in the MEF Key Registration Request in step 4 of clause 8.3.5.2.7.

If verification succeeds, then the MEF Client shall store the information in the MEF Key Registration Response and associate this information with the [*authenticationProfile*] MO node. The MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_OK.

If verification fails, or if procedure fails for any other reason, then the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_MO_NODE_PROCESSING_FAILED.

8.3.9.9.5 Process [*authenticationProfile*] MO Node with MAF-established symmetric key

Purpose: Processing an [*authenticationProfile*] MO node with MEF-established symmetric key SUID (in the set {31,32,33}) ensures that the MEF Client, acting as a Source MAF Client, establishes a symmetric key with the MAF for subsequent use with the [*authenticationProfile*] MO node.

Preconditions:

A. This procedure will succeed only if the [*authenticationProfile*] MO node has a child [*MAFClientRegCfg*] MO node and a parent MO Node which may be of type [*registration*] or [*dataCollection*].

B. This procedure assumes that the MEF Client, acting as an MAF Client, has a currently-valid MAF Client Registration with the MAF and administrating stakeholder identified in the child [*MAFClientRegCfg*] MO node.

Procedure: The MEF Client shall attempt the MAF Key Registration procedure as described in clause 8.8.2.7, with the MAF Client acting as the Source MAF Client and with following clarifications.

At step 4 in clause 8.8.2.7, the MEF Client shall form the MAF Key Registration Request (see table 8.8.2.7-1) as follows:

- *MAF-FQDN*: shall be set to the value of the *fqdn* attribute in the child [*MAFClientRegCfg*] MO node;
- *expirationTime* shall be computed by adding the current time to the value of the *keyRegDuration* in the [*authenticationProfile*] MO node;
- *labels* shall be assigned the value of the *keyRegLabels* attribute in the [*authenticationProfile*] MO node;
- *adminFQDN*: shall be assigned to the value of the *adminFQDN* attribute in the child [*MAFClientRegCfg*] MO node;
- *SUID*: shall be assigned to the value of the *SUID* attribute in the child [*authenticationProfile*] MO node;
- *targetIDs*: shall be assigned according to the parent MO node of the [*authenticationProfile*] MO node:
  - In the case of a parent [*registration*] MO node, the *targetIDs* shall be assigned the CSE-ID of the Registrar CSE.
  - In the case of a parent [*dataCollection*] MO node, the *targetIDs* shall be assigned the CSE-ID determined from the *containerPath* attribute of the [*dataCollection*] MO node.
- *keyValue*: shall not be present.

The MEF Client shall select a protocol (HTTP, CoAP, WebSocket) based on the protocol(s) supported by the MAF Client, and the protocol(s) supported by the MAF is indicated by the presence of the *httpPort*, *coapPort* and *websocketPort* in the child [*MAFClientRegCfg*] MO node. The MEF Client shall use the port number provided in the appropriate *httpPort*, *coapPort* or *websocketPort* attribute of the child [*MAFClientRegCfg*] MO node.

At step 10 in clause 8.8.2.7, If the MEF Client receives a successful MAF Key Registration Response (see table 8.8.2.7-2) then the MEF Client shall verify the following information received in the MAF Key Registration Response:

- *expirationTime:* Verification fails if this value is prior to the time when verification is performed.
- *Source MEF Client ID:* Verification fails if this value is distinct from the identifier of the MEF Client.
- *adminFQDN*: Verification fails if this value is distinct from the corresponding value sent in the MEF Key Registration Request in step 4 of clause 8.8.2.7.
- *SUID*: Verification fails if this value is distinct from the corresponding value sent in the MEF Key Registration Request in step 4 of clause 8.8.2.7.
- *targetIDs*: Verification fails if this value is distinct from the corresponding value sent in the MEF Key Registration Request in step 4 of clause 8.8.2.7.

If verification succeeds, then the MEF Client shall store the information in the MEF Key Registration Response and associate this information with the [*authenticationProfile*] MO node. The MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_OK.

If verification fails, or if procedure fails for any other reason, then the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_MO_NODE_PROCESSING_FAILED.

8.3.9.9.6 Process [*authenticationProfile*] MO Node with Certificate

Purpose: Processing an [*authenticationProfile*] MO node with Certificate SUID (in the set {41,42,43}) ensures that the MEF Client has access to a local copy of the MEF Client's Certificate and corresponding private key, for subsequent use with the [*authenticationProfile*] MO node.

Preconditions:

A. This procedure will succeed only if the Node has been provisioned with the certificate matching the *myCertFingerprint* attribute in the [*authenticationProfile*] MO node.

Procedure: The MEF Client obtains the value of the *myCertFingerprint* attribute in the [*authenticationProfile*] MO node. The MEF Client determines if it has a local copy of a certificate corresponding private key with the certificate matching the *myCertFingerprint* attribute corresponding private key.

- If a local copy of the certificate and corresponding private key are not present, then the MEF Client associates the certificate and corresponding private key with the [*authenticationProfile*] MO node for subsequent use. The MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_MO_NODE_PROCESSING_FAILED.

- If a local copy of the certificate and corresponding private key are present, then the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_OK.

8.3.9.9.7 [*trustAnchorCred*]-specific Processes

Purpose: Processing a [*trustAnchorCred*] MO node ensures that the MEF Client has a local copy of the trust anchor CA certificate identified by the [*trustAnchorCred*] MO node.

Elements of *objectTypeSpecificArgs*: The *objectTypeSpecificArgs* element is not present for the [*trustAnchorCred*] specialization.

Processing a [*trustAnchoCred*] MO Node:

1) The MEF Client retrieves the value of the *certFingerprint* attribute of the [*trustAnchorCred*] MO Node. The MEF Client determines if the MEF Client has a local copy of a certificate matching the *certFingerprint* attribute:

   - If a local copy of the certificate is present, then the MEF Client associates the certificate with the [*trustAnchorCred*] MO node for subsequent use. The MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_OK.

   - If a local copy of the certificate is not present, then the MEF Client proceeds to step 2.

2) The MEF Client obtains the value of the *URI* attribute of the [*trustAnchorCred*] MO Node. The MEF Client attempts to retrieve the trust anchor certificate by performing a HTTPS GET procedure towards the *URI*:

   - If the HTTPS GET procedure is not successful, then the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_MO_NODE_PROCESSING_FAILED.

- If the HTTPS GET procedure is successful, then the MEF Client extracts the payload. The MEF Client parses the payload to determine if it is a certificate, and if parsing succeeds, then the MEF Client verifies that the received certificate matches the *certFingerprint* attribute of the [*trustAnchorCred*] MO Node:

    - If parsing succeeds and the received certificate matches the *certFingerprint* attribute of the [*trustAnchorCred*] MO Node, then the MEF Client associates the certificate with the [*trustAnchorCred*] MO node for subsequent use. The MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_OK.

    - Otherwise, the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_MO_NODE_PROCESSING_FAILED.

8.3.9.9.8 [*MAFClientRegCfg*]-specific Processes

Purpose: Processing an [*MAFClientRegCfg*] MO node ensures that the MEF Client, acting as an MAF Client, has successfully registered with the MAF using the attributes in the [*MAFClientRegCfg*] MO node.

Elements of *objectTypeSpecificArgs*: The *objectTypeSpecificArgs* element is not present for the [*MAFClientRegCfg*] specialization.

Processing an [*MAFClientRegCfg*] MO Node: The MEF Client shall attempt the MAF Client Registration procedure as described in clause 8.8.2.3, with the MAF Client acting as the Source MAF Client and with following clarifications:

At step 2 in clause 8.8.2.3, the MEF Client shall form the MAF Client Registration Request (see table 8.8.2.3-1) from the attributes of the [*MAFClientRegCfg*] MO node as follows:

- *MAF-FQDN*: shall be set to the value of the *fqdn* attribute;
- *expirationTime* shall be assigned to the value of *expirationTime* attribute;
- *labels* shall be assigned the value of the *labels* attribute;
- *adminFQDN*: shall be assigned to the value of the *adminFQDN* attribute;

The MEF Client shall select a protocol (HTTP, CoAP, WebSocket) based on the protocol(s) supported by the MAF Client, and the protocol(s) supported by the MAF is indicated by the presence of the *httpPort*, *coapPort* and *websocketPort* in the child [*MAFClientRegCfg*] MO node. The MEF Client shall use the port number provided in the appropriate *httpPort*, *coapPort* or *websocketPort* attribute of the child [*MAFClientRegCfg*] MO node.

At step 3 in clause 8.8.2.3, If the MEF Client receives a successful MEF Key Registration Response (see table 8.8.2.3-2) then the MEF Client shall verify the following information received in the MEF Key Registration Response:

- *expirationTime:* Verification fails if this value is prior to the time when verification is performed.
- *MAF Client ID:* Verification fails if this value is distinct from the identifier of the MEF Client.
- *adminFQDN*: Verification fails if this value is distinct from the corresponding value sent in the MEF Key Registration Request in step 2 of clause 8.8.2.3.

If verification succeeds, then the MEF Client shall store the information in the MEF Key Registration Response and associate this information with the [*MAFClientRegCfg*] MO node. The MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_OK.

If verification fails, or if the procedure fails for any other reason, then the MEF Client exits the procedure, triggering the MEF Client Command Update procedure for this *cmdID* with *cmdStatusCode* set to the value for MEF_CLIENT_CMD_MO_NODE_PROCESSING_FAILED.

8.4 End-to-End Security of Primitives (ESPrim)

8.4.1 Purpose of E2E Security of Primitives (ESPrim)

End-to-End Security of Primitives (ESPrim) provides an interoperable framework for securing oneM2M primitives so CSEs (forwarding the primitive) do not need to be trusted with the confidentiality and integrity of the primitive. ESPrim provides mutual authentication, confidentiality, integrity protection and a freshness guarantee (bounding the age of ESPrim Objects).

Applicable use cases and requirements are discussed in ETSI TR 118 512 [i.16].

The present document assumes that the ESPrim end-points are the Originator and Receiver of the primitive.

The present document specifies credential management aspects and data protection aspects for ESPrim. Transport of ESPrim Objects is described in ETSI TS 118 101 [1].

8.4.2 End-to-End Security of Primitives (ESPrim) Architecture

The credential management aspects and data protection aspects for ESPrim are specified in the present clause. Clause 11.3.2, ETSI TS 118 101 [1] specifies the transport of ESPrim Objects.

The primitive to be secured is called the *inner primitive*, and the primitive which is used to transport a secured inner primitive is called the *outer primitive*. The inner primitive is protected using encryption and integrity protection which takes a symmetric key *sessionESPrimPrimKey* as input. The *sessionESPrimKey* is derived from a *pairwiseESPrimKey*, established between the Originator and Receiver, and a *receiverESPrimRandObject* and *originatorESPrimRandObject*.

Sequence of events for ESPrim consists of three main phases:

1) Establishing pairwiseESPrimKey.

2) Establishing sessionESPrimKey at the Originator.

3) Securing a primitive exchange.

Figure 8.4.2-1 shows the ESPrim message flow for establishing *pairwiseESPrimKey* and *sessionESPrimKey* at the Originator. Figure 8.4.2-2 shows the ESPrim message flow for securing a Primitive Exchange.

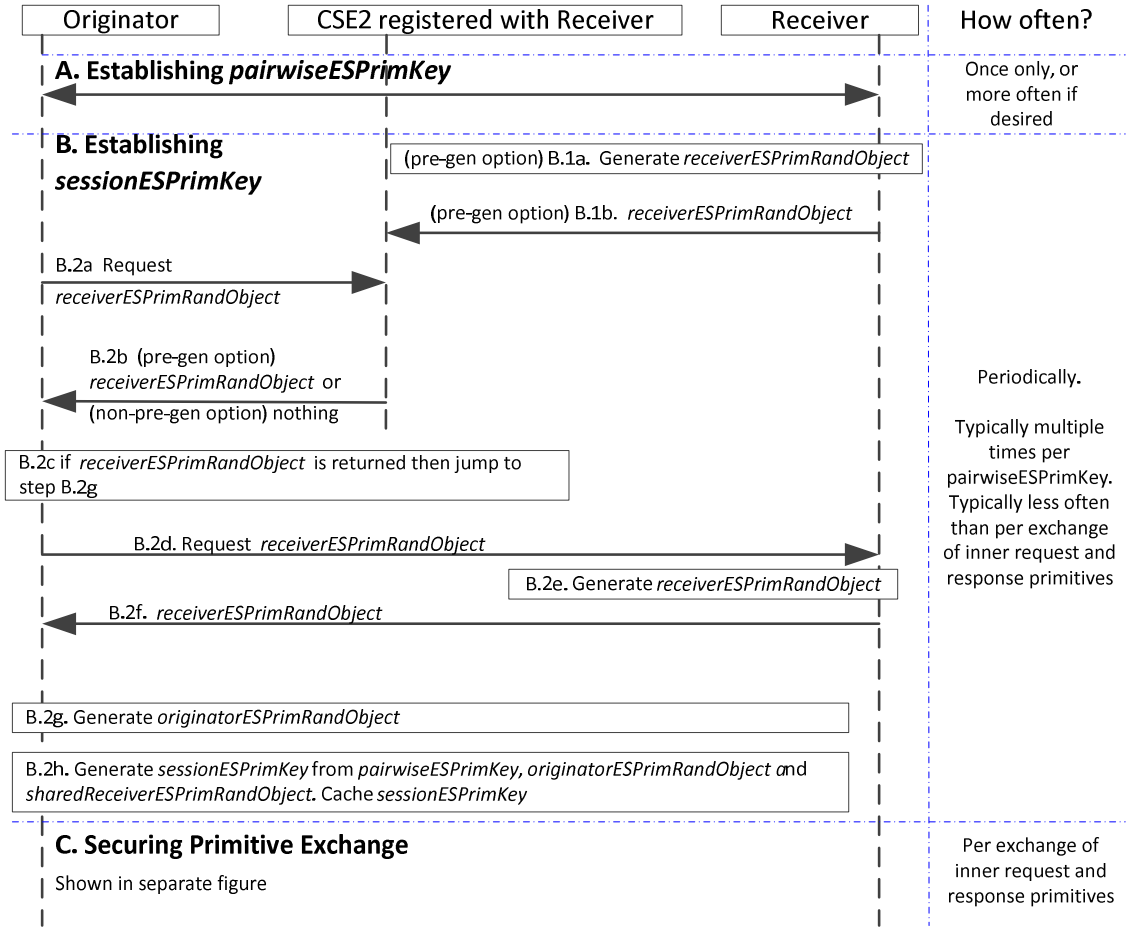


Figure 8.4.2-1: Message flow for establishing *pairwiseESPrimKey* and establishing *sessionESPrimKey* at the Originator in the End-to-End Security of Primitives (ESPrim) Procedure

A. Establishing *pairwiseESPrimKey*: The *pairwiseESPrimKey* may be established using either of the following frameworks:

- **Provisioned** *pairwiseESPrimKey* **Framework:** The Originator and Receiver shall be provisioned with *pairwiseESPrimKey*. This credential shall be provisioned via one of:
  - pre-provisioning;
  - a Remote Security Provisioning Frameworks (RSPF), specified in clause 8.3; or
  - End-to-End Security Certificate based Key Establishment (ESCertKE) between the Originator and Receiver, specified in clause 8.7.

  The Originator and Receiver also establish *pairwiseESPrimKeyID* and optionally *pairwiseESPrimKeyLifetime* during this process. If no *pairwiseESPrimKeyLifetime*, is provided, then then *pairwiseESPrimKey* never expires. The Originator and Receiver cache the (*pairwiseESPrimKeyID*, *pairwiseESPrimKey*, (optional) *pairwiseESPrimKeyLifetime*) object to use for processing subsequent primitives.

- **MAF ESPrim Framework:** The Originator and M2M Authentication Function (MAF) authenticate each other using symmetric keys (which may be pre-provisioned or remotely provisioned) and derive a M2M Secure Connection key (Kc) and corresponding key identifier (KcID). The Originator generates *pairwiseESPrimKey* from Kc and a reserved string. The value of KcID is used in phase C as the *pairwiseESPrimKeyID* in the JWE/XML-ENC object. The Originator caches the (*pairwiseESPrimKeyID*, *pairwiseESPrimKey*) pair to use for processing subsequent primitives. The Receiver retrieves Kc and a credential lifetime from the MAF after it receives an inner request primitive secured using the corresponding *pairwiseESPrimKey*' see step C.6.a.

  When *pairwiseESPrimKey* is established using the MAF option, then it typically has a shorter lifetime than the former option.

- **Receiver indicates support for ESPrim:** If Receiver supports ESPrim, the Receiver shall ensure the following for the Receiver's <*remoteCSE*> resource on all CSEs which are registered with the Receiver:

  - The Receiver's <*remoteCSE*> resource shall include the *e2eSecInfo* attribute.

  - The *e2eSecInfo* attribute in this resource shall indicate support for ESPrim.

B. Establishing *sessionESPrimKey* at the Originator: The Receiver shall select to either (a) pre-generate a *receiverESPrimObject* which is distributed for use by multiple Originators for establishing *sessionESPrimKey*, or (b) generate a unique *receiverESPrimRandObject* upon request (in which case no action is required prior to receiving such a request). If the Receiver selects to generate a unique *receiverESPrimRandObject* upon request, then In the latter case, the Receiver shall ensure that the *sharedReceiverESPrimRandObject* parameter is not present in the *e2eSecInfo* attribute in the Receiver's <*remoteCSE*> resource on all CSEs which are registered with the Receiver. The absence of the *sharedReceiverESPrimRandObject* parameter indicates that the Receiver will provide a unique *receiverESPrimRandObject* upon request.

B.1 **(If the Receiver selected to use pre-generation) Receiver pre-generation of *sharedReceiverESPrimRandObject***: If the Receiver selected to pre-generate and distribute a *receiverE2SPrimandObject*, the Receiver performs the following steps every time the Receiver wishes to provide a new shared *receiverESPrimRandObject*

B.1a.1 The Receiver shall generate a receiverESPrimRandObject including the following parameters:

- The Receiver shall generate a 128-bit fresh random value *ESPrimRandValue*.

- The Receiver shall assign *ESPrimRandExpiry*, indicating when the *receiverESPrimRandObject* shall cease to be valid.

- The Receiver shall assign an *ESPrimRandID* for the *receiverESPrimRandObject* which shall satisfy the following criteria: (a) the *ESPrimRandID* shall indicate that the *receiverESPrimRandObject* is shared; (b) the *ESPrimRandID* shall be unique among the shared *receiverESPrimRandObject* issued by the Receiver and valid at the time of issuance. These criteria ensure that the *receiverESPrimRandObject* can be uniquely identified until it expires.

- The Receiver shall include a list of *sessionESPrimKeyGenerationAlgorithmID* values identifying the algorithms that the Receiver is willing to use for generating *sessionESPrimKey* using this *receiverESPrimRandObject*.

- The Receiver shall include a list of *AEADAlgorithmID* values identifying the AEAD algorithms that the Receiver is willing to use with this *receiverESPrimRandObject*.

B.1b The Receiver shall update the *sharedReceiverESPrimRandObject* parameter of the *e2eSecInfo* attribute in the Receiver's <*remoteCSE*> resource on all CSEs which are registered with the Receiver.

NOTE 1: At a given point in time, multiple Originators will be using identical values for the current *sharedReceiverESPrimRandObject*.

B.2 **Originator obtaining *receiverESPrimRandObject***: The Originator shall perform the following steps whenever the Originator establishes a *sessionESPrimKey* with the Receiver:

B.2a The Originator shall perform a Retrieve on the *e2eSecInfo* attribute in the Receiver's <*remoteCSE*> resource on a CSE, here denoted CSE2, which is registered with the Receiver.

B.2b If the *e2eSecInfo* attribute is present in the Receiver's <*remoteCSE*> resource on CSE2, then CSE2 shall return the *e2eSecInfo* attribute. Otherwise CSE2 returns an error message to the Originator.

B.2c The Originator determines if the Receiver supports ESPrim, which requires that the *e2eSecInfo* attribute is present and the *e2eSecInfo* attribute indicates support for ESPrim.

B.2.c.1 If the Receiver does not support ESPrim, then the Originator shall abort the procedure.

B.2c.2 If the Receiver supports ESPrim, and the *e2eSecInfo* attribute includes a *sharedReceiverESPrimRandObject* parameter, then the Originator examines the *ESPrimRandExpiry* in this parameter to determine if the *sharedReceiverESPrimRandObject* has expired. If the *sharedReceiverESPrimRandObject* has not expired, then the Originator shall set *receiverESPrimRandObject* to the value of *sharedReceiverESPrimRandObject* and proceeds to step B.2g. If the *sharedReceiverESPrimRandObject* has expired, then the Originator shall proceed to step B.2d.

B.2c.3 If the Receiver supports ESPrim, and the *e2eSecInfo* attribute does not include a *sharedReceiverESPrimRandObject* parameter, then the Originator shall proceed to step B.2d.

B.2d The Originator shall send a message to the Receiver requesting a receiverESPrimRandObject.

B.2e The Receiver, upon receiving such a request, shall generate a receiverESPrimRandObject including the following parameters:

- The Receiver shall generate a 128-bit fresh random value *ESPrimRandValue*.
- The Receiver shall assign *ESPrimRandExpiry*, indicating when the *receiverESPrimRandObject* shall cease to be valid.
- The Receiver shall assign an *ESPrimRandID* for the *receiverESPrimRandObject* which shall satisfy the following criteria: (a) the *ESPrimRandID* shall indicate that the *receiverESPrimRandObject* is not shared; (b) the *ESPrimRandID* shall be unique among the non-shared *receiverESPrimRandObject* issued by the Receiver and valid at the time of issuance. These criteria ensure that the *receiverESPrimRandObject* can be uniquely identified until it expires.
- The Receiver shall include a list of *sessionESPrimKeyGenerationAlgorithmID* values identifying the algorithms that the Receiver is willing to use for generating *sessionESPrimKey* using this *receiverESPrimRandObject*.
- The Receiver shall include a list of *AEADAlgorithmID* values identifying the AEAD algorithms that the Receiver is willing to use with this *receiverESPrimRandObject*.

B.2f The Receiver shall send a message to the Originator with the *receiverESPrimRandObject*.

B.2g The Originator shall generate an *originatorESPrimRandObject* including the following parameters:

- The Originator shall generate a 128-bit fresh random value *ESPrimRandValue*.
- The Originator shall assign *ESPrimRandExpiry*, indicating when the *originatorESPrimRandObject* shall cease to be valid. The *ESPrimRandExpiry* shall not be later than the *ESPrimRandExpiry* in the *receiverESPrimRandObject* obtained in step B.2c or B.2f.
- The Originator shall assign an *ESPrimRandID* for the *originatorESPrimRandObjectID* which shall satisfy the following criteria: (a) the *ESPrimRandID* shall indicate that the *originatorESPrimRandObject* is not shared; (b) the *ESPrimRandID* shall be unique among the non-shared *originatorESPrimRandObject* issued by the Originator and valid at the time of issuance. These criteria ensure that the *originatorESPrimRandObject* can be uniquely identified until it expires.
- The Originator shall include a single *sessionESPrimKeyGenerationAlgorithmID* identifier for the algorithm that the Originator is applying for generating *sessionESPrimKey* using this *originatorESPrimRandObject*. This shall be one of the algorithms identified by the *sessionESPrimKeyGenerationAlgorithmID* values in *receiverSPrimERandObject* obtained in step B.2c or B.2f.
- The Originator shall include a list of *AEADAlgorithmID* values identifying the AEAD algorithms that the Originator is willing to use with this *originatorESPrimRandObject*. This shall be one or more of the algorithms identified by *AEADAlgorithmID* in *receiverESPrimRandObject* obtained in step B.2c or B.2f.

B.2h The Originator shall generate the *sessionESPrimKey* from the *pairwiseESPrimKey*, r*eceiverESPrimRandObject* received at step B.2c and *originatorESPrimRandTuple* generated at step B.2.g.

NOTE 2: The *sessionESPrimKey* used to secure an inner request primitive is always used to protect the corresponding inner response primitive, so *sessionESPrimKey* has to be cached at least until the corresponding inner response primitive is expected to have been received. The Originator typically caches the *sessionESPrimKey* for a longer period of time since the *sessionESPrimKey* can be used for securing multiple primitive exchanges.

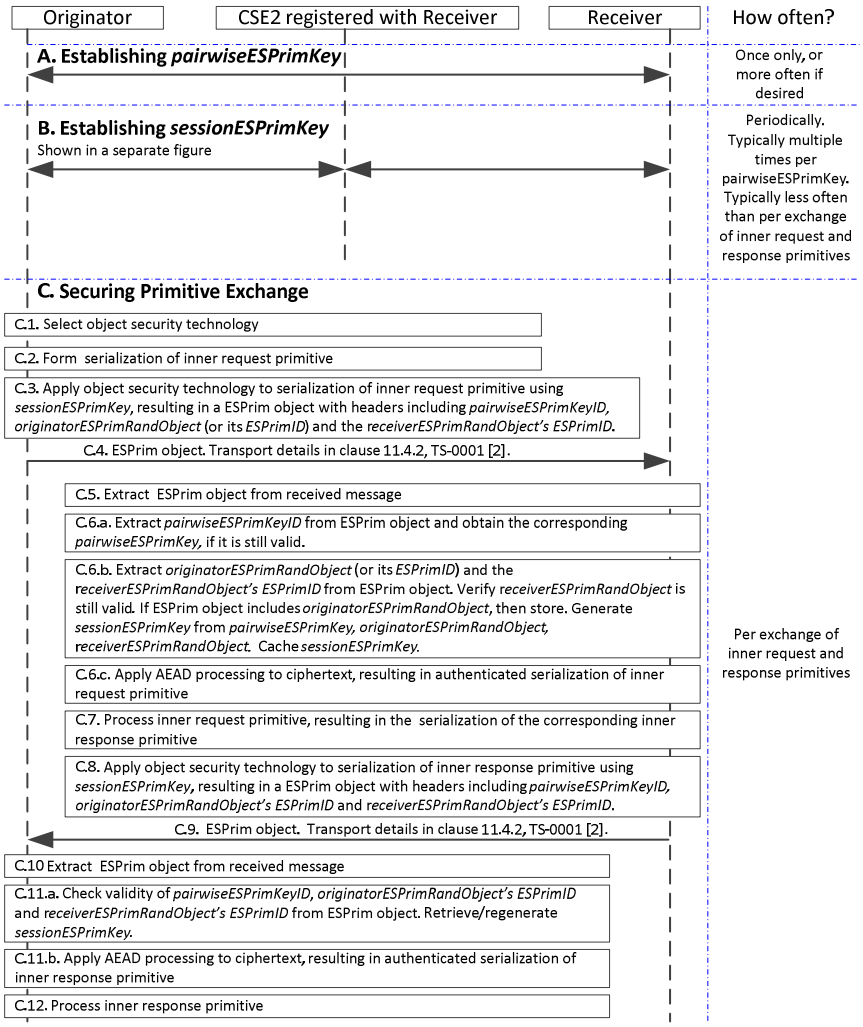


Figure 8.4.2-2: Message flow for securing a primitive exchange in the End-to-End Security of Primitives (ESPrim) Procedure

C. Securing a Primitive Exchange:

NOTE 3: The Originator selects the type of serialization (e.g. JSON, XML) of the inner request primitive to be secured.

C.1 The Originator selects the object security technology depending on the object security technology supported by the Originator, and the type of serialization of the inner request primitive.

C.2 The Originator shall form the serialization of the inner request primitive.

C.3 The Originator shall produce a ESPrim Object by applying the object security technology as follows:

- One or more headers of the a ESPrim Object shall include the following information:
  - *pairwiseESPrimKeyID*.
  - *originatorESPrimRandObject* used to generate the *sessionESPrimKey*, or the corresponding *ESPrimRandID*. If there is the possibility that this ESPrim Object could be the first ESPrim Object received by the Receiver which is secured by the Originator using a specific *originatorESPrimRandObject*, then the full *originatorESPrimRandObject* shall be included. Otherwise, one of *originatorESPrimRandObject* or *ESPrimRandID* shall be included.
  - *ESPrimRandID* of the *receiverESPrimRandObject* used to generate the *sessionESPrimKey*.
  - *AEADAlgorithmID* for the ESPrim Object. This shall be one of the AEAD algorithms identified in *originatorESPrimRandObject*.

- The plaintext (to be encrypted) shall be the serialization of the inner request primitive.
- The *sessionESPrimKey* shall be used directly as the symmetric key providing authenticated encryption of the plaintext, resulting in the ciphertext in the ESPrim Object. The ciphertext is assumed to include the MIC for verifying integrity of the inner request primitive.

C.4 The Originator shall form an outer request primitive for transporting the ESPrim Object as described in in ETSI TS 118 101 [1]. The Originator shall send the outer request primitive to the Receiver.

C.5 The Receiver processes the received outer request primitive to extract the ESPrim Object as described in in ETSI TS 118 101 [1].

C.6 The Receiver shall process the ESPrim Object:

C.6a The Receiver shall extract the *pairwiseESPrimKeyID* from the ESPrim Object headers and obtain the corresponding *pairwiseESPrimKey*:

If the *pairwiseESPrimKeyID* is of the form for a Provisioned *pairwiseESPrimKey*, then the Receiver shall use the corresponding (previously-provisioned) *pairwiseESPrimKey*.

If the *pairwiseESPrimKeyID* is of the form for a MAF *pairwiseESPrimKey*: If this this is the first time that the Receiver received a message with this *pairwiseESPrimKeyID,* then the following process shall be performed.

C.6a.1 The Receiver shall identify the MAF from the *pairwiseESPrimKeyID* (which is a KcID).

C.6a.2 The Receiver shall establish a secure TLS connection to the MAF and request the M2M Secure Connection key (Kc) and Kc Lifetime corresponding to *pairwiseESPrimKeyID* (which is identical to KcID).

C.6a.3 The MAF shall provide Kc and Kc Lifetime to the Receiver.

C.6a.4 The Receiver shall generate the *pairwiseESPrimKey* from Kc and a reserved string.

C.6a.5 The Receiver shall set *pairwiseESPrimKeyLifetime* to Kc Lifetime.

C.6a.6 The Receiver shall cache (p*airwiseESPrimKeyID, pairwiseESPrimKey, pairwiseESPrimKeyLifetime)* for use for processing subsequent primitives.

If the Receiver has previously cached (*pairwiseESPrimKeyID*, pairwiseESPrimKey, pairwiseESPrimKeyLifetime), and pairwiseESPrimKeyLifetime has not yet expired, then the Receiver may use the cached pairwiseESPrimKey.

C.6b The Receiver shall apply the following process to generate the *sessionESPrimKey*:

C.6b.1 The Receiver shall extract *ESPrimRandID* of the *receiverESPrimRandObject* from the headers of the ESPrim object, and attempt to retrieve the corresponding cached value of *receiverESPrimRandObject*. If no cached value is found, or the cached value is expired, then the Receiver shall respond to the outer request primitive with an error.

C.6b.2 The Receiver shall extract the encoding of the *originatorESPrimRandObject* or *ESPrimRandID* of the *originatorESPrimRandObject* from the headers of the ESPrim object, and apply the appropriate decoding. If an *originatorESPrimRandObject* is provided then it shall be cached. If an *ESPrimRandID* is provided then the Receiver shall retrieve the corresponding cached value of *originatorESPrimRandObject*.If no cached value is found, or the cached value is considered expired, then the Receiver shall respond to the outer request primitive with an error message.

The Receiver shall process the *originatorESPrimRandObject*:

C.6b.2.i The Receiver shall check the *ESPrimExpiry* in the *originatorESPrimRandObject* to verify (a) that this Expiry is not already in the past and (b) the *ESPrimExpiry* is not later than the *ESPrimExpiry* in the *receiverESPrimRandObject*.

C.6b.2.ii The Receiver shall extract *sessionESPrimKeyGenerationAlgorithmID* and verify that the identified algorithm matches one of the *sessionESPrimKeyGenrationAlgorithmID* in *receiverESPrimRandObject*.

C.6b.2.iii The Receiver shall generate the *sessionESPrimKey* from the *pairwiseESPrimKey*, r*eceiverESPrimRandObject* and *originatorESPrimRandObject* or retrieve the value of *sessionESPrimKey* if previously generated and cached.

NOTE 4: The *sessionESPrimKey* used to secure an inner request primitive is always used to protect the corresponding inner response primitive, so *sessionESPrimKey* has to be cached at least until the corresponding inner response primitive is sent. The Receiver typically caches the *sessionESPrimKey* for a longer period of time since the originator can use the *sessionESPrimKey* for securing multiple primitive exchanges.

C.6c Authenticated decryption steps at the Receiver:

C.6c.1 The Receiver shall extract *AEADAlgorithmIDs* in *originatorESPrimRandObject* and verify that the identified set of algorithms is a subset of the set in *AEADAlgorithmIDs* in *receiverESPrimRandObject*.

The Receiver shall process the *AEADAlgorithmID* in the ESPrim Object headers and verify that the identified algorithm matches one of the *AEADAlgorithmIDs* in *originatorESPrimRandObject*.

C.6c.2 The Receiver shall apply the AEAD Algorithm identifier in the ESPrim Object header to the ciphertext parameter in the ESPrim Object resulting in verified plaintext, using the *sessionESPrimKey*. The ciphertext is assumed to include the MIC for verifying integrity of the inner request primitive. The authenticated serialization of the inner request primitive is the verified plaintext output by the AEAD algorithm.

C.7 The Receiver shall process the inner request primitive, resulting in a serialization of the corresponding inner response primitive.

NOTE 5: Steps C.2 to C.7 are mirrored closely by C.8 to C.12, with the Originator and Receiver swapping their participation in the exchange, and the request primitives replaced by response primitives. There are minor differences: in particular some of the request processing in steps C.2 to C.7 is not required in the response processing since the Originator has already generated *sessionESPrimKey;* it is only necessary to identify the appropriate *sessionESPrimKey*, as performed in step C.11.a.

C.8 The Receiver shall use the same *sessionESPrimKey* as used in the ESPrim Object received at step C.5. Consequently, *pairwiseESPrimKeyID*, *originatorESPrimRandObject* and *receiverESPrimRandObject* are the same as for the received at step C.5.

The Receiver shall produce an ESPrim Object by applying the object security technology as follows:

- One or more headers of the a ESPrim Object shall include the following information:
  - *pairwiseESPrimKeyID*
  - *originatorESPrimRandObject's ESPrimRandID.*
  - *receiverESPrimRandObject's ESPrimRandID*
  - *AEADAlgorithmID* for the ESPrim Object. This shall be one of the AEAD algorithms identified in *originatorESPrimRandObject*.
- The plaintext (to be encrypted) shall be the serialization of the inner response primitive.
- The *sessionESPrimKey* shall be used directly as the symmetric key providing authenticated encryption of the plaintext, resulting in the ciphertext in the ESPrim Object. The ciphertext is assumed to include the MIC for verifying integrity of the inner request primitive.

C.9 The Receiver shall form an outer request primitive for transporting the ESPrim Object as described in in ETSI TS 118 101 [1]. The Originator shall send the outer response primitive to the Receiver.

C.10 The Originator processes the received outer response primitive to extract the ESPrim Object as described in ETSI TS 118 101 [1].

C.11 The Originator shall process the ESPrim Object.

C.11a The Originator shall extract, from the headers of the ESPrim object, the values of *pairwiseESPrimKeyID, originatorESPrimRandObject's ESPrimRandID, receiverESPrimRandObject's ESPrimRandID*. These values shall match the *pairwiseESPrimKeyID, originatorESPrimRandObject's ESPrimRandID, receiverESPrimRandObject's ESPrimRandID* of a session that the Originator considers to be currently valid.

If any of these values have expired, then the outer response primitive shall be discarded.

NOTE 6: For this reason, the expiry of these values need to be great enough to allow receiving the corresponding inner response primitive.

Otherwise, the Originator shall use the cached value of *sessionESPrimKey* corresponding to these values, or may regenerate *sessionESPrimKey*.

C.11b The Originator shall apply the AEAD Algorithm identified in the ESPrim Object header to the ciphertext parameter in the ESPrim Object resulting in verified plaintext, using *sessionESPrimKey*. The ciphertext is assumed to include the MIC for verifying integrity of the inner request primitive. The authenticated serialization of the inner request primitive is the verified plaintext output by the AEAD algorithm.

C.12 The Originator shall process the inner response primitive.

8.4.3 End-to-End Security of Primitives (ESPrim) Protocol Details

8.4.3.1 End-to-End Security of Primitives (ESPrim) Parameter Definitions

8.4.3.1.1 originatorESPrimRandObject parameter definition

The structure of the originator2ERandObject parameter is shown in table 8.4.3.1.1-1. This parameter is used in establishing *sessionESPrimKey* as part of End-to-End Security of Primitives (ESPrim), described in clause 8.4.2. The data type of the originatorESPrimRandObject parameter is specified in ETSI TS 118 104 [4].

Table 8.4.3.1.1-1: Structure of the *originatorESPrimRandObject* parameter

Element Path	Multiplicity	Description
esprimRandID	1	An identifier for the originatorESPrimRandObject, assigned by the CSE or AE generating the originatorESPrimRandObject.
esprimRandValue	1	A 128-bit randomly-generated value.
esprimRandExpiry	1	Time when the originatorESPrimRandObject expires.
esprimKeyGenAlgID	1	The enumerated identifier of the algorithm selected for sessionESPrimKey generation by the CSE or AE generating the originatorESPrimRandObject..
esprimProtocolAndAlgIDs	1	A list of enumerated identifiers for AEAD Algorithms supported by the CSE or AE generating the originatorESPrimRandObject.

8.4.3.1.2 receiverESPrimRandObject parameter definition

The structure of the receiver2ERandObject parameter is shown in table 8.4.3.1.2-1. This parameter is used in establishing sessionESPrimKey as part of End-to-End Security of Primitives (ESPrim), described in clause 8.4.2. The data type of the receiverESPrimRandObject parameter is specified in ETSI TS 118 104 [4].

Table 8.4.3.1.2-1: Structure of the *receiverESPrimRandObject* parameter

Element Path	Multiplicity	Description
esprimRandID	1	An identifier for the receiverESPrimRandObject, assigned by the CSE or AE generating the receiverESPrimRandObject
esprimRandValue	1	A 128-bit randomly-generated value
esprimRandExpiry	1	Time when the receiverESPrimRandObject expires
esprimKeyGenAlgIDs	1	A list of enumerated identifiers for algorithms supported for sessionESPrimKey generation by the CSE or AE generating the receiverESPrimRandObject
esprimProtocolAndAlgIDs	1	A list of enumerated identifiers for AEAD Algorithms supported by the CSE or AE generating the receiverESPrimRandObject

8.4.3.1.3 *e2eSecInfo* resource attribute definition

The *e2eSecInfo* attribute occurs in the <*CSEBase*>, <*remoteCSE*> and <*AE*> resource types. The structure of the *e2eSecInfo* resource attribute is shown in table 8.4.3.1.3-1. This parameter is used in establishing sessionESPrimKey as part of End-to-End Security of Primitives (ESPrim), described in clause 8.4.2. The data types are specified in ETSI TS 118 104 [4].

Table 8.4.3.1.3-1: Structure of the *e2eSecInfo* attribute

Element Path	Multiplicity	Description
supportedE2ESecurityFeatures	1	A list of Security Usage Identifiers (SUIDs) for the End-to-End Security Features supported by the CSE or AE associated with the <*CSEBase*>, <*remoteCSE*> or <*AE*> resource containing the *e2eSecInfo* resource attribute.
e2ECertificates	0..1	A list of certificates associated with the CSE or AE associated with the <*CSEBase*>, <*remoteCSE*> or <*AE*> resource containing the *e2eSecInfo* resource attribute.
sharedReceiverESPrimRandObject	0..1	A receiverESPrimRandObject parameter (see clause 8.4.3.1.2) generated by the CSE or AE associated with the <*CSEBase*>, <*remoteCSE*> or <*AE*> resource containing the *e2eSecInfo* resource attribute.

8.4.3.2 ESPrim Object formatting and processing using the JWE Compact Serialization

Background: JSON Web Encryption (JWE) specified in IETF RFC 7516 [50], provides a simple format for encrypting any data object. Two JWE serializations are provided: a compact, URI-safe serialization, and a JSON serialization.

The JW End-to-End Security of Primitives (ESPrim) Compact Serialization necessary formatting and parsing can be easily encoded without generic tools for formatting or parsing JSON.

In the JWE Compact Serialization, a JWE is represented as the concatenation of five JWE parameters:

BASE64URL(UTF8(JWE Protected Header)) || '.' ||
BASE64URL(JWE Encrypted Key) || '.' ||
BASE64URL(JWE Initialization Vector) || '.' ||
BASE64URL(JWE Ciphertext) || '.' ||
BASE64URL(JWE Authentication Tag)

where BASE64URL(OCTETS) denotes the base64url encoding of OCTETS, per section 2 of the JSON Web Signature specification IETF RFC 7515 [51]. Base64 and base64url encodings are defined in IETF RFC 4648 [41].

NOTE 1: If OCTETS is an empty octet sequence, then IETF RFC 7515 [51] defines BASE64URL(OCTETS) to be the empty string.

NOTE 2: The JWE Compact Serialization is not as flexible as the JWE JSON serialization, however the JWE compact serialization provides sufficient flexibility for ESPrim Objects. Moreover ease of formatting and parsing JWE Compact Serialization provides a simple solution for both XML and JSON representations of primitives.

JWE Parameter definitions for ESPrim: Table 8.4.3.2-1, specifies these values of the five JWE parameters when the JWE Compact Serialization.

Table 8.4.3.2-1: JWE Components used in ESPrim Objects

JWE value	element type	Empty	Component Content
JWE Protected Header	JSON	Never	See table 8.4.3.2-2 JWE Protected Header Parameters
JWE Encryption Key	Binary value	Always	This value is empty for the key management mode used for ESPrim
JWE Initialization Vector	Binary value	Conditional	As per IETF RFC 7516 [50]
JWE Ciphertext	Binary value	Never	
JWE Authentication Tag	Binary value	Conditional	
NOTE: Whether these components are empty or not is conditional on the algorithm selected for encryption.			

Table 8.4.3.2-2 "JWE Protected Header Parameters" describes the parameters in the JWE Protected Header when using JWE for ESPrim.

Table 8.4.3.2-2: JWE Protected Header Parameters

Element path	Multiplicity for ESPrim	Purpose	Specification of element	Description of assigned value
`"alg"`	1	Key management mode	[50]	`"dir"` indicating Direct Encryption.
`"enc"`	1	Encryption Algorithm		The options available here are the same as for ESData use of JWE. See clause 8.7.3.
`"kid"`	1	Key identifier	[50]	Identifier for pairwiseESPrimKey.
`"cty"`	0..1	Media type of the secured Content	[50]	Dictated by the serialization of the primitive (XML or JSON) selected by the Originator.
`"ori"`	1	Identify originators input to session key generation	Clause 8.4.3.1.1	esprimRandID of the originatorESPrimRandObject used to generate sessionESPrimKey for this ESPrim Object.
`"rri"`	1	Identify receiver's input to session key generation	Clause 8.4.3.1.2	esprimRandID of the receiverESPrimRandObject used to generate sessionESPrimKey for this ESPrim Object.
`"oro"`	0..1	originatorESPrimRandObject	Clause 8.4.3.1.1 using JSON serialization	JSON representation of an originatorESPrimRandObject generated by the Originator. Sent only by the Originator.
`"rro"`	0..1	receiverESPrimRandObject	Clause 8.4.3.1.2 using JSON serialization	JSON representation of an receiverESPrimRandObject generated by the Receiver. Sent only by the Receiver.

The following text provides further explanation of the parameters in the JWE Protected Header when using JWE for ESPrim.

Recall that an ESPrim Object is formed by encrypting the inner primitive using the symmetric key sessionESPrimKey. The generation of sessionESPrimKey from a pairwiseESPrimKey, originatorESPrimRandObject and receiverESPrimRandObject is separate from the key management available through JWE - from the perspective of JWE, sessionESPrimKey is simply a secret symmetric key value shared between Originator and Receiver. JWE uses the Direct Encryption key management mode in this scenario. The JWE serialization will be required to transport the pairwiseESPrimKey identifier, the esprimRandIDs of the originatorESPrimRandObject and receiverESPrimRandObject, and optionally a receiverESPrimRandObject or originatorESPrimRandObject so that the Originator and Receiver can generate the correct sessionESPrimKey.

- The "alg" (Algorithm) JWE header parameter [50] shall be set to the value of "dir" to indicate Direct Encryption key management mode.
- The "enc" (Encryption Algorithm) JWE header parameter [50] may be selected by the sender of the ESPrim Object. The encryption algorithm shall agrees with the esprimProtocolAndAlgIDs in identified originatorESPrimRandObject and the esprimProtocolAndAlgIDs in the identified receiverESPrimRandObject.
- The "kid" (Key ID) JWE header parameter [50] shall be set to the pairwiseESPrimKey identifier.
- The "cty" shall identify the media type of the representation (JSON or XML) of the inner primitive.

- The esprimRandIDs of the originatorESPrimRandObject and receiverESPrimRandObject shall be communicated in the "ori" (Originator Rand ID) and "rri" (Receiver Rand ID) parameters included in the JWE Protected header. These parameters are specific to oneM2M and shall conform to the definition of the esprimRandID in clauses 8.4.3.1.1 and 8.4.3.1.2.

- The originator may include a originatorESPrimRandObject in an "oro" (Originator Rand Object) parameter - either at the beginning of an ESPrim session or to refresh the session keys "in-band" during an existing ESPrim session. In the former case, "ori" shall match the esprimRandID in the originatorESPrimRandObject, but this restriction does not apply in the latter case. This parameter is defined in clauses 8.4.2 and 8.4.3.1.1, and the JSON representation shall be used.

- The receiver may include a receiverESPrimRandObject in an "rro" (Receiver Rand Object) - to refresh the session keys "in-band" during an existing ESPrim session. This parameter is defined in clauses 8.4.2 and 8.4.3.1.2, and the JSON representation shall be used.

Forming an ESPrim Object: The inner primitive representation shall be encrypted as specified for message encryption in [50], with the following clarification:

1) The JWE Protected Header shall be composed as described above.

   1.1) The Content Encryption Key (CEK) shall be the sessionESPrimKey generated using the pairwiseESPrimKey, originatorESPrimRandObject and receiverESPrimRandObject identified in the JWE Protected Header.

2) The plaintext shall be the inner primitive representation.

3) The JWE Initialization Vector, JWE Ciphertext and JWE Authentication Tag shall be generated from the Protected JWE Header, the plaintext and CEK as specified in IETF RFC 7516 [50], according to the identified encryption algorithm

4) The JWE Compact Serialization shall be composed from the JWE parameters.

The ESPrim Object is the JWE Compact Serialization.

Processing an ESPrim Object: The JWE Compact Serialization shall be processed as specified for message decryption in [50], with the following clarification:

1) The JWE Protected Header shall be composed as described above.

   1.1) If originatorESPrimRandObject or receiverESPrimRandObject is included, then these shall be recorded.

   1.2) The Content Encryption Key (CEK) shall be the sessionESPrimKey generated using the pairwiseESPrimKey, originatorESPrimRandObject and receiverESPrimRandObject identified in the JWE Protected Header. The generation of SessionESPrimKey is specified in clause 8.4.2.

2) The plaintext shall be generated from the JWE Initialization Vector, JWE Ciphertext, JWE Authentication Tag and CEK as specified in IETF RFC 7516 [50], according to the identified encryption algorithm.

8.5 End-to-End Security of Data (ESData)

8.5.1 Purpose of ESData

End-to-End Security of Data (ESData) provides an interoperable framework for protecting data that ends up transported using oneM2M reference points, in order that so transited CSEs do not need to be trusted with that data. The data to be protected is called the *ESData Payload*. ESData payload could typically compose all or part of an attribute value (e.g. *content* attribute value of a <*contentInstance*> resource) or a primitive parameter (e.g. a signed, self-contained access token communicated in a request primitive to obtain dynamic authorization).

NOTE: Within the scope of clause 8.5, terms including the word "ESData" can be shortened by removing "ESData" to facilitate readability. For example, "ESData Payload" is often shortened to "Payload".

Applicable use cases and requirements are discussed in ETSI TR 118 512 [i.16].

ESData assumes that there is a single *ESData Source End-Point* applying the ESData processing to the Payload to obtain an *ESData Envelope* containing the secured data and necessary headers, with one or more *ESData Target End-Points* applying the ESData processing to the Envelope to extract the verified data. The Payload is composed of plaintext (which is to be encrypted and integrity protected) and associated authenticated data (which is to be integrity protected only).

There is no inherent restriction on which entities can be Source End-points and Target End-Points; these end-points may be entities inside a oneM2M system (that is, AEs and CSEs) or entities outside of a oneM2M system (for example, entities which are part of a system that interworks with oneM2M).

The present document specifies credential management aspects and data protection aspects for ESData. The present document does not address transporting ESData Envelopes.

8.5.2 ESData Architecture

8.5.2.1 List of ESData Security Classes and ESData Protection Options

The following ESData security classes are provided:

- **Encryption only:** (see note) which offers confidentiality and integrity protection. This payload is protected using symmetric keys, and these symmetric keys are established using one or more of the following:
  - Symmetric keys otherwise established with the target end-points. In this case, the source end-point can be authenticated - unless the symmetric key was shared with multiple target end-points.
  - Target end-points certificate. When target end-point certificate are used, the target end-point cannot authenticate the source end-point.

NOTE: Strictly speaking, this class provides encryption and integrity protection, but this name aligns usage in protocols such as JSON Web Encryption (JWE) and XML-Encryption which can provide both encryption and integrity protection.

- **Signature only:** which offers source authentication, integrity protection and (when asymmetric digital signatures are used) non-repudiation. This uses either symmetric keys based MIC or asymmetric digital signatures verified using source end-point certificates.
- **Nested Sign-then-Encrypt:** This is used in cases where encryption is required in addition to source authentication and/or non-repudiation using a source end-point certificate. A digital signature(s) on the payload is signed first, and then encryptions is applied to combination of the payload and digital signature.

ESData supports using multiple credentials for protecting a single payload unit.

Each ESData Security class supports three ESData protection options, as shown in table 8.5.2.1-1.

Table 8.5.2.1-1: ESData protection options

ESData Security Class	ESData Protection Option	Key Management	Source Verification	Non-Repudiation
Encryption only (clause 8.5.2.2)	Encryption using Provisioned Symmetric ESData Key	Provisioned Symmetric Key	Symmetric	-
	Encryption using TEF	TEF	Symmetric	-
	Encryption using Target End-Point Certificate	Certificate	-	-
Signature only (clause 8.5.2.3)	MIC using Provisioned Symmetric ESData Key	Provisioned Symmetric Key	Symmetric	-
	MIC using TEF	TEF	Symmetric	-
	Digital Signature using End-Point Source End-Point Certificate	Certificate	Certificate	Certificate
Nested-Sign-then Encrypt (clause 8.5.2.4)	Digital Signature using End-Point Source End-Point Certificate followed by any combination of Encryption-Only Protection Options	Provisioned Symmetric Key(s) and/or TEF(s) and/or Certificate(s) for Encryption. Certificate for Signature	Certificate	Certificate

8.5.2.2 Encryption-Only ESData Security Class

8.5.2.2.1 Encryption-Only ESData Security Class Overview

The ESData protection option supported for the Encryption-Only Security Class are listed in table 8.5.2.1-1 "ESData protection Options".

Encryption-Only ESData supports encrypting using any combination of Protection Options and using multiple credentials for each protection option.

ESData Encryption Mode. ESData Security Class supports two main encryption modes:

- **ESData Direct Encryption Mode:** In this mode, a symmetric key is used directly in the Encryption algorithm for securing the payload. The Direct Encryption mode is recommended only in scenarios meeting the following criterions:
  - Scenarios in which minimizing overhead of data objects is a very high priority.
  - The Encryption function will not be used with the same key value more than 2^{32} times, at least for AES GCM, for the reasons discussed in clause 8.4 of IETF RFC 7518 [49].

  This mode shall only be used when there is a single symmetric key being used to protect the payload.
- **ESData Encrypted Key Mode:** In this mode, the Content Encryption Key (CEK) (used in the Encryption algorithm for securing the payload) is encrypted using one or more credentials and the encrypted CEK is provided in a header with the secured data.

Encryption Mode Applicability Constraints. In scenarios where either:

- Encryption using Provisioned Symmetric ESData Key is applied using a single provisioned symmetric key; or
- Encryption using TEF is applied using a single TEF-registered symmetric key;
- Then either Direct Encryption Mode or Encrypted Key Mode may be applied.

In all other scenarios, Encrypted Key Mode shall be applied.

High Level Sequence of Events. The following text describes the sequence of events when using an Encryption-Only Security Class.

NOTE: The present document does not describe the processes by which the Source End-Point and Target End-Point(s) decide on the credentials to be used for securing a payload, and the encryption algorithm to be applied.

A. Credential Configuration: The Source End-Point obtains the credentials needed to secure the payload for the intended Target End-Point(s). This can include any combination of the Protection Options, multiple credentials allowed for each Protection Options:

- **Encryption using Provisioned Symmetric ESData Key:** The Source End-Point and Target End-Point(s) are provisioned with Provisioned Symmetric ESData Key as described in clause 8.5.2.2.2.
- **Encryption using TEF:** The Source End-Point generates a random secret TEF-registered symmetric key, and registers this key with the TEF as described in clause 8.5.2.2.3.
- **Encryption using Certificates:** The Source End-Point obtains the certificate of the Target End-Point as described in clause 8.5.2.2.4.

B. Source End-Point CEK Management:

- If Direct Encryption Mode is to be applied, then the Provisioned Symmetric ESData Key or Registered TEF Symmetric Key shall be used directly as CEK. The use of Direct Encryption Mode shall be indicated in the *ESData Headers*: header parameters of the ESData Envelope. The Provisioned Symmetric ESData Key or Registered TEF Symmetric Key shall be identified in the headers.

- Otherwise, the Source End-Point shall generates a random secret value for the Content Encryption Key CEK and shall encrypt CEK using the credential(s) obtained in Phase A "Credential Management", as described in clauses 8.5.2.2.2, 8.5.2.2.3 and 8.5.2.2.4. Each encrypted CEKs shall be added to the Headers, along with the identifier for the credential which is to be used to decrypt the encrypted CEK. The CEK value may be used for a single payload or may be used for multiple payloads.

C. Source End-Point Encryption:

C.1 The Encryption algorithm shall be identified in the Headers.

C.2 The Source End-Point shall apply the encryption process for the identified algorithm to the payload using CEK. The plaintext is encrypted to form the ciphertext, and the combination of plaintext and associated Authenticated Data (AAD) is integrity protected by the generated MIC.

C.3 The Source End-Point shall form the Envelope from the Headers, ciphertext, AAD and MIC; this process may include encoding data using, for example, base64.

The present document does not specify how the Envelope is obtained or provided to the Target End Point(s). The following steps are applied at each Target End-Point.

D. Target End-Point CEK Management:

D.1 The Target End-Point parses the Envelope, applying any necessary encoding, and extracts the Header parameters.

D.2 If Direct Encryption Mode is indicated in the Headers, then the Target End-Point shall use the credential identifiers in the Headers to obtain the identified Provisioned Symmetric ESData Key or Registered TEF Symmetric Key (as described in clauses 8.2.2.2 or 8.5.2.2.3 respectively). The Target End-Point shall use this symmetric key directly as CEK.

D.3 Otherwise, the Target End-Point shall use the credential identifiers in the Headers to identify an encrypted CEK that can be decrypted by a credential known or available to the Target End-Point. The Target End-Point shall obtain that credential and decrypt the encrypted CEK as described in clauses 8.5.2.2.2 (Provisioned Symmetric ESData Key case), 8.5.2.2.3 (TEF case) and 8.5.2.2.4 (Target End-Point Certificate case). The Target shall use the resulting CEK for processing the secured payload of the Envelope. The Target End-Point may cache the CEK value due to the possibility of that CEK value being used to protect subsequent payloads.

E. Target End-Point Decryption:

E.1 The Target End-Point shall determine the appropriate Encryption algorithm identified in the Headers.

E.2 The Target End-Point shall apply the Encryption decryption process for the identified algorithm to the ciphertext, AAD and MIC using CEK, outputting the verified plaintext and verified AAD.

8.5.2.2.2 Encryption using Provisioned Symmetric ESData Key

For this Protection Option, the Source End-Point and each Target End-Point shall be provisioned with Provisioned Symmetric ESData Key, Provisioned Symmetric ESData Key Identifier and optionally Provisioned Symmetric ESData Key lifetime. This credential shall be provisioned via one of:

- Pre-provisioning;
- A Remote Security Provisioning Frameworks (RSPF), specified in clause 8.3; or
- Certificate based End-to-End Security Key Establishment between the Originator and Receiver, specified in clause 8.7.

8.5.2.2.3 Encryption using Trust Enabling Function

This is specified in clause 8.6.

8.5.2.2.4 Encryption using Target End-Point Certificates

8.5.2.2.4.1 Associating Public Key Certificate with Target End-Points

For this Protection Option, each Target End-Point shall be provisioned with a public key certificate which the Source End-Point trusts to be associated with the intended Target End-Point. The following options are supported:

- The Target End-Point Certificates may use the following Public Key Certificate flavours identified in clause 8.1.2.1:
  - In the case of a Raw Public Key Certificate, the Source End-Point shall be securely configured (either directly or remotely) to associate the Target End-Point with the raw public key or its hash. The details of this configuration are not provided in the present document.
  - In the case of a Device Certificate:
    - The Source End-Point shall be securely configured with the trust anchor in the certificate chain of the Device Certificate; typically during initial provisioning.
    - The Source End-Point shall be securely configured to associate the Target End-Point with the globally unique hardware instance identifier. The details of this configuration are not provided in the present document.
  - In the case of an AE-ID certificate or CSE-ID certificate, the Source-End-Point shall be securely configured with the trust anchor in the certificate chain of the AE-ID certificate or CSE-ID certificate; typically during initial provisioning. The Source End-Point then trusts that the Target End-Point with a particular AE-ID or CSE-ID is associated with the certificate that contains that AE-ID or CSE-ID.
  - In the case of a Node-ID certificate, the Source-End-Point shall be securely configured with the trust anchor in the certificate chain of the Node-ID certificate; typically during initial provisioning. The Source End-Point then trusts that the Target End-Point with a particular Node-ID is associated with the certificate that contains that Node-ID.
- The Target End-Point Certificates may use other Public Key Infrastructures, particularly when the Target End-Point is in a non-oneM2M system interworking with the oneM2M system. The present document provides no interoperability guarantees when such certificates are used.

Public keys for verifying signature cannot be used for this Protection Option.

8.5.2.2.4.2 Obtaining Target End-Point Certificates

The Source End-Point is unable to secure a message to the Target End-Point before obtaining the Target End-Point's certificate. The present document does not mandate the mechanism by which the Target End-Point's certificate is provided to the Source End-Point, and there are a variety of mechanisms which are suitable. The *e2ESecurityParameters* is a mechanism provided by oneM2M to allow the Source End-Point to retrieve certificates associated with a CSE or AE.

A Target End-Point AE may make certificates available at the *e2ESecurityParameters* attribute of the <*AE*> resource representing that AE. This retrieval process is not a reliably-secure mechanism for associating the Target End-Point with the certificate; clause 8.5.2.2.4.1.

A Target End-Point CSE may make certificates available at the *e2ESecurityParameters* attribute of the <*CSE*> and <*remoteCSE*> resources representing that CSE. This retrieval process is not a reliably-secure mechanism for associating the Target End-Point with the certificate; clause 8.5.2.2.4.1 shall also be applied.

8.5.2.3 Signature-Only ESData Security Class

8.5.2.3.1 Signature-Only ESData Security Class Overview

The ESData protection option supported for the Signature-Only ESData Security Class are listed in table 8.5.2.1-1 "ESData protection Options".

NOTE 1: The present document supports only one Signature-Only ESData Protection Option, but the clause is structured to support additional Signature-Only ESData Protection Options if desired in the future.

Signature-Only ESData supports encrypting using any combination of Protection Options and using multiple credentials for each protection option.

High Level Sequence of Events. The following text describes the sequence of events when using a Signature-Only Security Class.

NOTE 2: The present document does not describe the processes by which the Source End-Point and Target End-Point(s) decide on the credentials to be used for signing a payload, and the algorithms to be applied.

A. Credential Configuration: The Source End-Point obtains the credentials needed to sign the payload for the intended Target End-Point(s). This can include any combination of the Protection Options, multiple credentials allowed for each Protection Options:

- **MIC using Provisioned Symmetric ESData Key:** The Source End-Point and Target End-Point(s) are provisioned with Provisioned Symmetric ESData Key as described in clause 8.5.2.2.2.
- **MIC using TEF:** The Source End-Point generates a random secret TEF-registered symmetric key, and registers this key with the TEF as described in clause 8.5.2.2.3.
- **Digital Signature using Source End-Point Certificates:** The Source End-Point selects a private key and corresponding Source End-Point Certificate as described in clause 8.5.2.3.2.

B. Source End-Point Signing:

B.1 The Payload is encoded, for example, using base 64.

B.2 For each credential, the Source End-Point shall generate array of data elements as follows:

B.2.i The Source End-Point shall form a Header, identifying the digital signature or MIC algorithm, and the credential which can be used by a Target End-Point to verify the digital signature or MIC. If required, the header is also encoded, for example using base64.

B.2.ii The Source End-Point shall generate a signature/MIC by applying the appropriate digital signature or MIC algorithm to the Payload and Header using the appropriate credential, and encoding, for example using base 64.

B.2.iii The Source End-Point shall form a data element from the Headers, Payload and signature/MIC.

B.3 The Source End-Point shall form Envelope from the encoded Payload and the array of data elements generated at step B.2.

The present document does not specify how the Envelope is obtained or provided to the Target End Point(s). The following steps are applied at each Target End-Point.

C. Target End-Point Verification:

C.1 The Target End-Point parses the Envelope, extracting the encoded Payload and the array of data elements, each containing a Header and a signature/MIC.

C.2 The Target End-Point shall examine the array of data elements to identify data elements which can be verified by a credential which may be trusted by the Target End-Point. For each such data element:

C.2.i The Target End-Point shall obtain the identified credential according to clauses 8.5.2.2.2 (Provisioned Symmetric ESData Key case), 8.5.2.2.3 (TEF case), and 8.5.2.3.2 (using Source End-Point Certificate case).

C.2.ii The Target End-Point shall verify the MIC or signature in using the credential.

C.3 The Target End-Point shall decode the verified encoded Payload - outputting the original Payload - and shall record the credential(s) used to verify the Payload.

8.5.2.3.2 Digital Signature using Source End-Point Certificate

8.5.2.3.2.1 Associating Public Key Certificate with Source End-Point

For this Protection Option, each Source End-Point shall be provisioned with a public key certificate which the Target End-Point trusts to be associated with the intended Source End-Point. The following options are supported:

- The Source End-Point Certificates may use the following Public Key Certificate flavours identified in clause 8.1.2.1:
  - In the case of a Raw Public Key Certificate, the Target End-Point shall be securely configured (either directly or remotely) to associate the Source End-Point with the raw public key or its hash. The details of this configuration are not provided in the present document.
  - In the case of a Device Certificate:
    - The Target End-Point shall be securely configured with the trust anchor in the certificate chain of the Device Certificate; typically during initial provisioning.
    - The Target End-Point shall be securely configured to associate the Source End-Point with the globally unique hardware instance identifier. The details of this configuration are not provided in the present document.
  - In the case of an AE-ID certificate or CSE-ID certificate, the Target-End-Point shall be securely configured with the trust anchor in the certificate chain of the AE-ID certificate or CSE-ID certificate; typically during initial provisioning. The Target End-Point then trusts that the Source End-Point with a particular AE-ID or CSE-ID is associated with the certificate that contains that AE-ID or CSE-ID.
  - In the case of a Node-ID certificate, the Target-End-Point shall be securely configured with the trust anchor in the certificate chain of the Node-ID certificate; typically during initial provisioning. The Target End-Point then trusts that the Source End-Point with a particular Node-ID is associated with the certificate that contains that Node-ID.
- The Source End-Point Certificates may use other Public Key Infrastructures, particularly when the Source End-Point is in a non-oneM2M system interworking with the oneM2M system. The present document provides no interoperability guarantees when such certificates are used.

Public keys for verifying signatures shall be used for this Protection Option.

8.5.2.3.2.2 Obtaining Source End-Point Certificates

The Target End-Point is unable to secure a message to the Source End-Point before obtaining the Source End-Point's certificate. The present document does not mandate the mechanism by which the Source End-Point's certificate is provided to the Target End-Point using any mechanism, and there are a variety of mechanisms which are suitable. The *e2ESecurityParameters* is a mechanism provided by oneM2M to allow the Target End-Point to retrieve certificates associated with a CSE or AE.

A Source End-Point AE may make certificates available at the *e2ESecurityParameters* attribute of the <*AE*> resource representing that AE. This retrieval process is not a reliably-secure mechanism for associating the Source End-Point with the certificate; clause 8.5.2.3.2.1 shall also be applied.

A Source End-Point CSE may make certificates available at the *e2ESecurityParameters* attribute of the <*CSE*> and <*remoteCSE*> resources representing that CSE. This retrieval process is not a reliably-secure mechanism for associating the Source End-Point with the certificate; clause 8.5.2.3.2.1 shall also be applied.

8.5.2.4 Nested Sign-then-Encrypt

For these options, the following high-level steps are performs (Credential Configuration steps and CEK Management steps are not shown):

1) The Source End-Point shall generate an inner Envelope containing one or more digital signatures for the inner Payload using one or more certificates according to the "Digital Signature using Source End-Point Certificate" Signature-Only Protection Option in clause 8.5.2.3.

2) The Source End-Point shall set the inner Envelope produced by step 1 to be the plaintext of the outer Payload which is then encrypted using any combination of Encryption-Only Protection Options in clause 8.5.2.2. This results in an outer Envelope.

The present document does not specify how the outer Envelope is obtained or provided to the Target End Point(s). The following steps are subsequently applied at each Target End-Point:

3) The Target End-Point shall decrypt the outer Envelope produced by step 1 using one of the Encryption-Only Protection Options in clause 8.5.2.2, resulting in the outer Payload which is also the inner Envelope.

4) The Target End-Point shall verify one or more digital signatures in the inner Envelope using one or more certificates according to the "Digital Signature using Source End-Point Certificate" Signature-Only Protection Option in clause 8.5.2.3, resulting in the verified inner Payload.

8.5.3 End-to-End Security of Data (ESData) Protocol Details

8.5.3.1 Introduction

The End-to-End Security of Primitives (ESData) security classes support protocols shown in table 8.5.3.1-1.

Table 8.5.3.1-1: ESData Security Classes, and mapping to XML-based and JSON-Based security protocols

ESData Security Class	XML-Based	JOSE: JSON-Based Security
Encryption only	XML-ENC applied to ESData payload	JWE applied to ESData payload
Signature only	XML-SIG applied to ESData payload	JWS applied to ESData payload
Nested-Sign-then-Encrypt	XML-SIG applied to ESData payload, with XML-ENC applied to the result	JWS applied to ESData payload, with JWE applied to the result

The JOSE option allows a flexible JSON Serializations in addition to less flexible Compact Serializations (which are also URI safe). So there are three serialization options: XML, JWE/JWS using JSON Serialization and JWE/JWS using Compact Serializations.

8.5.3.2 Encryption-Only ESData Security Class Protocol Details

To maintain consistency, key management algorithms are provided which are available in both XML-ENC [55] and JSON Web Encryption (JWE) [50]:

- Direct Encryption.
- AES Key Wrap, using 128-bit, 256-bit keys.
- RSA-OAEP with MGF1 with SHA256.
- Elliptic Curve Diffie-Hellman (ECDH) Key Agreement in Ephemeral-Static Mode using AES-Key Wrap.

Table 8.5.3.2-1 identifies the key management algorithms that are supported in XML-Encryption for the Encryption-only ESData Security Class.

Table 8.5.3.2-1: Key management algorithms that are supported in XML-Encryption for Encryption-only ESData Security Class

Key Management	Algorithm		<xenc:EncryptionMethod Algorithm=".."> for encrypting the key	Other parameters
Direction Encryption	n/a		n/a	<ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#"> <ds:KeyName>John Smith</ds:KeyName>
Symmetric Key Wrap	AES Key Wrap, with	128-bit key	http://www.w3.org/2001/04/xmlenc#kwaes128	
		192-bit key	http://www.w3.org/2001/04/xmlenc#kwaes192	
		256-bit key	http://www.w3.org/2001/04/xmlenc#kwaes256	
RSA	RSA-OAEP with MFG1 and SHA-256		http://www.w3.org/2009/xmlenc11#rsa-oaep	<xenc11:MGF Algorithm="http://www.w3.org/2009/xmlenc11#mgf1sha256>
ECDH Key Agreement	ECDH-ES with AES Key Wrap	128-bit key	http://www.w3.org/2001/04/xmlenc#kwaes128	<xenc:AgreementMethod Algorithm="http://www.w3.org/2009/xmlenc11#ECDH-ES">
		192-bit key	http://www.w3.org/2001/04/xmlenc#kwaes192	
		256-bit key	http://www.w3.org/2001/04/xmlenc#kwaes256	

Table 8.5.3.2-2 identifies the payload encryption algorithms that are supported in XML-Encryption for the Encryption-only ESData Security Class.

Table 8.5.3.2-2: Payload encryption algorithms that are supported in XML-Encryption for Encryption-only ESData Security Class

Payload Encryption Algorithm		<EncryptionMethod Algorithm="..">
AES GCM with	128-bit key	http://www.w3.org/2009/xmlenc11#aes128gcm
	192-bit key	http://www.w3.org/2009/xmlenc11#aes192gcm
	256-bit key	http://www.w3.org/2009/xmlenc11#aes256gcm

The output generated by XML-Encryption is serialized as an XML object. The XML-Encryption object may be transported "plain" – with no encoding, or may be encoded in base64.

Table 8.5.3.2-3 identifies the key management algorithms that are supported in JWE for the Encryption-only ESData Security Class.

Table 8.5.3.2-3: Key management algorithms that are supported in JSON Web Encryption (JWE) for Encryption-only ESData Security Class

Key Management	Algorithm		"alg":".."
Direction Encryption	n/a		dir
Symmetric Key Wrap	AES Key Wrap, with	128-bit key	A128KW
		192-bit key	A192KW
		256-bit key	A256KW
RSA	RSA-OAEP with MFG1 and SHA-256		"alg": "RSA-OAEP-256"
ECDH Key Agreement	ECDH-ES with AES Key Wrap	128-bit key	ECDH-ES+A128KW
		192-bit key	ECDH-ES+A192KW
		256-bit key	ECDH-ES+A256KW

Table 8.5.3.2-4 identifies the payload algorithms that are supported in JWE for the Encryption-only ESData Security Class.

Table 8.5.3.2-4: Payload encryption algorithms that are supported in JSON Web Encryption (JWE) for Encryption-only ESData Security Class

Payload Encryption Algorithm		"enc":".."
AES GCM with	128-bit key	A128GCM
	192-bit key	A192GCM
	256-bit key	A256GCM

The output generated by JWE conforms to either the JWE JSON Serialization or a URI-safe JWE Compact Serialization. The JWE JSON Serialization may be transported "plain" - with no encoding, or may be encoded in base64. ETSI TS 118 104 [4] defines the datatype m2m:e2eCompactJWE for the JWE Compact Serialization.

8.5.3.3 Signature-Only ESData Security Class Protocol Details

To maintain consistency, signature types are provided which are available in both XML-Signature [52] and JSON Web Signature (JWS) [51].

- HMAC using SHA-256, SHA-384 or SHA-512.
- RSA signature using PKCS1-v1.5 or PSS and MGF1 with SHA-256, SHA-384 or SHA-512.
- ECDSA signature using P-256, P-384 or P-512 with SHA-256, SHA-284 or SHA-512 respectively.
- ECDSA signature using FRP256v1 and brainpoolP256r1 curves [74] with SHA-256 for both curves.

Table 8.5.3.3-1 identifies the algorithms that are supported in XML-SIG for Signature-only ESData Security Class.

Table 8.5.3.3-1: Algorithms that are supported in XML-Signature for Signature-only ESData Security Class

Signature Type	Algorithm		<SignatureMethod Algorithm="..">
HMAC	SHA-256		http://www.w3.org/2001/04/xmldsigmore#hmacsha256
	SHA-384		http://www.w3.org/2001/04/xmldsigmore#hmacsha384
	SHA-512		http://www.w3.org/2001/04/xmldsigmore#hmacsha512
RSA	RSA PKCS1-v1.5 or PSS and MGF1 with:	SHA-256	http://www.w3.org/2001/04/xmldsigmore#rsasha256
		SHA-384	http://www.w3.org/2001/04/xmldsigmore#rsasha384
		SHA-512	http://www.w3.org/2001/04/xmldsigmore#rsasha512
ECDSA	P-256 and SHA-256		http://www.w3.org/2001/04/xmldsigmore#ecdsasha256
	P-384and SHA-384		http://www.w3.org/2001/04/xmldsigmore#ecdsasha384
	P-512 and SHA-512		http://www.w3.org/2001/04/xmldsigmore#ecdsasha512
	FRP256v1 and SHA-256		See [74]
	brainpoolP256r1 and SHA-256		See [75]

The XML-Signature object may be transported "plain" - with no encoding, or may be encoded in base64.

Table 8.5.3.3-2 identifies the algorithms that are supported in JWS for Signature-only ESData Security Class.

Table 8.5.3.3-2: Algorithms that are supported in JSON Web Signature (JWS) for Signature-only ESData Security Class

Signature Type	Algorithm		"alg":".."
HMAC	SHA-256		HS256
	SHA-384		HS384
	SHA-512		HS512
RSA	RSA PKCS1-v1.5 or PSS and MGF1 with:	SHA-256	RS256
		SHA-384	RS384
		SHA-512	RS512
ECDSA	P-256 and SHA-256		ES256
	P-384and SHA-384		ES384
	P-512 and SHA-512		ES512

The output generated by JWS conforms to either the JWS JSON Serialization or a URI-safe JWS Compact Serialization. The JWS JSON Serialization may be transported "plain" – with no encoding, or may be encoded in base64. ETSI TS 118 104 [4] defines the datatype m2m:e2eCompactJWS for the JWS Compact Serialization.

8.5.3.4 Nested-Sign-then-Encrypt ESData Security Class Protocol Details

The high level steps for the Nested-Sign-then-Encrypt ESData Security Class are described in clause 8.5.2.4. The inner Envelope shall be generated and processed according to one or more RSA or ECDSA signature types specified in clause 8.5.3.3. The inner Envelope shall be generated and processed according to any combination of one or more key management algorithms specified in clause 8.5.3.2.

8.6 Remote Security Frameworks for End-to-End Security

8.6.1 Overview on Remote Provisioning and Registration of Credentials for End-to-End Security

8.6.1.1 Introduction

The Remote Provisioning Framework for End-to-End Security shall involve the ability for an entity to register and provision end-to-end credentials by means of a Trust Enabling Function for end-to-end security. An M2M Enrolment Function, M2M Authentication Function or a MN-CSE that is equipped with the ability to register and provision end-to-end security credentials may act as a Trust Enabling Function for end-to-end security.

The End-to-End Security Credentials derived may be used for providing the following security protection mechanisms:

- Message (primitive) integrity and authenticity using a Message Integrity Code (MIC).
- Message (primitive) confidentiality.
- Integrity and authenticity of the data (attributes) using Data Integrity Tag (DIT).
- Confidentiality of data (attributes).

Security protected messages and data (attributes) may be enveloped using ESPrim and ESData respectively using mechanisms described in clause 8.4 and 8.5. Message authenticity/integrity and confidentiality are provided using ESPrim, while integrity and confidentiality of application Data (attributes) are provided by using ESData Objects.

End-to-End Security may be provided using:

1) Leveraging Remote Security Provisioning process based on clause 8.3 and described in clause 8.6.2.
2) Using Source-generated Credentials described in clause 8.6.3.

8.6.1.2 Overall Description of Registration and Remote Provisioning for End-to-End Security

This clause provides description of mechanisms that may be employed for generation, registration and provisioning of credential(s) that shall be used for end-to-end security. Based on security requirements or security profile associated with an Entity (e.g. AE) and indicated within the <*e2ESecurityCapabilities*> resource described in clause 9.6.1.3.2 in ETSI TS 118 101 [1], appropriate end-to-end security credentials shall be generated. The remote provisioning mechanisms leverages the mechanisms described in clause 8.3 on the Remote Security Provisioning Frameworks and extends the mechanism in order that end-to-end Security credentials may be registered and shall be provisioned for entities that are more than one-hop away from one another. Figure 8.6.1.2-1 provides a sequence of high-level steps that may be followed for remote registration and provisioning of end-to-end credentials.

The steps involved in end-to-end security protection involve:

1) A Source ESP End-Point identifying the right set of security mechanisms and generating appropriate credentials.
2) Registering the credentials with a Trust Enabling Function.
3) The TEF provisions end-to-end credentials to a Target ESP End-Point.
4) Processing of ESData/ESPrim using the end-to-end credentials.

When a Remote Security Provisioning process is used, then steps 1) and 2) shall be primarily performed by a TEF. In the case, where Source-generated end-to-end security credentials are used, then steps 1) and 2) shall be performed by the Source ESF End-Point.

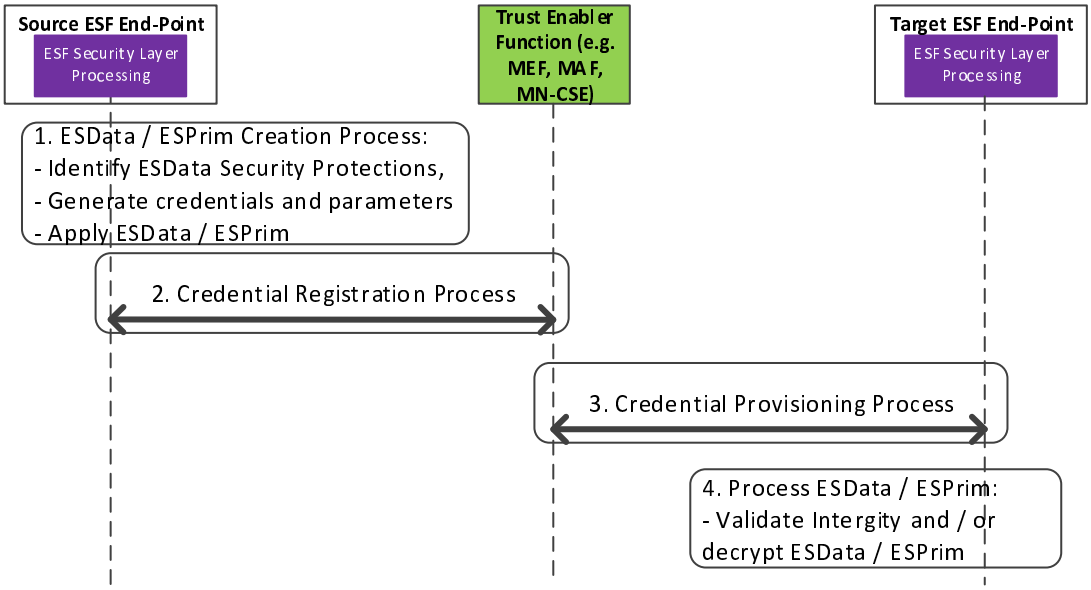


Figure 8.6.1.2-1: High-level summary of Credential Registration and Provisioning Process

The end-to-end credential registration and provisioning process for providing ESData/ESPrim involves the following steps:

- Creation of ESData/ESPrim by the Source ESF End-Point Process involves:

  a) Identification of security protection mechanisms based on the security requirements associated with the application data.

  b) Based upon the security requirements, appropriate security credentials and associated parameters are generated.

  c) The application data is then protected using the security credential(s) and associated parameters in order to generate the ESData/ESPrim.

NOTE 1: In the case of Remote Security Provisioning process, these steps a) and b) are performed by a Trust Enabling Function. Whereas in the case of Source-generated, the above described steps are followed by the Source.

- Credential Registration Process:

  a) The Source ESF End-Point registers the credential(s) and associated parameters with a Trust Enabling Function.

  b) The Source ESP End-Point shall provide the identity of the Target ESF End-Point(s) that is authorized to be provisioned with the end-to-end credentials and associated parameters.

NOTE 2: In the case of Remote Security Provisioning process, the Credential Registration process is performed by a Trust Enabling Function. Whereas in the case of Source-generated, the above described steps are followed by the Source.

- Credential Provisioning/Requisition Process:

  a) A Target ESF End-Point may request ESData/ESPrim credentials by using a Credential-Id that was obtained as part of the ESData/ESPrim.

  b) Based on the authorization information provided as part of the Credential Registration Process and using the Credential-Id, the Trust Enabling Function provisions the appropriate credentials and associated cryptographic parameters to the authenticated and authorized Target ESF End-Point.

- Process the ESData/ESPrim:

  a) The Target ESF End-Point uses the credentials provisioned by the Trust Enabling Function in order to process the ESData/ESPrim.

b) Processing of ESData/ESPrim would involve the integrity verification/authentication of the application data and/or decryption of the data and messages respectively.

8.6.2 Remote Security Provisioning Process for End-to-End Security Credentials

This clause describes the Remote Provisioning of Symmetric End-to-End Security credentials. The end-to-end security credentials shall be generated after having completed the Remote Provisioning of symmetric credentials using the Provisioned Symmetric Key or the MAF-based Symmetric Key Security Association Establishment Processes as described in clause 8.3.

Based on the higher-level requirements, appropriate end-to-end credentials may be generated using Remote Security Provisioning process by using pre-provisioned credentials. Illustrated in figure 8.6.2-1 is a high-level key generation process.

As part of the "End-to-End Key Generation" mechanism, the enrolee and the enrolment target generate end-to-end credentials using the Kpsa as the master key in order to generate the end-to-end master key. If the Enrolee is an AE (Source ESF End-Point), and the Enrolment Target is a CSE (Target ESF End-Point), then an end-to-end master credential, Ke2e_master is generated. An Example of end-to-end key generation using IETF RFC 5869 [48] is provided below.

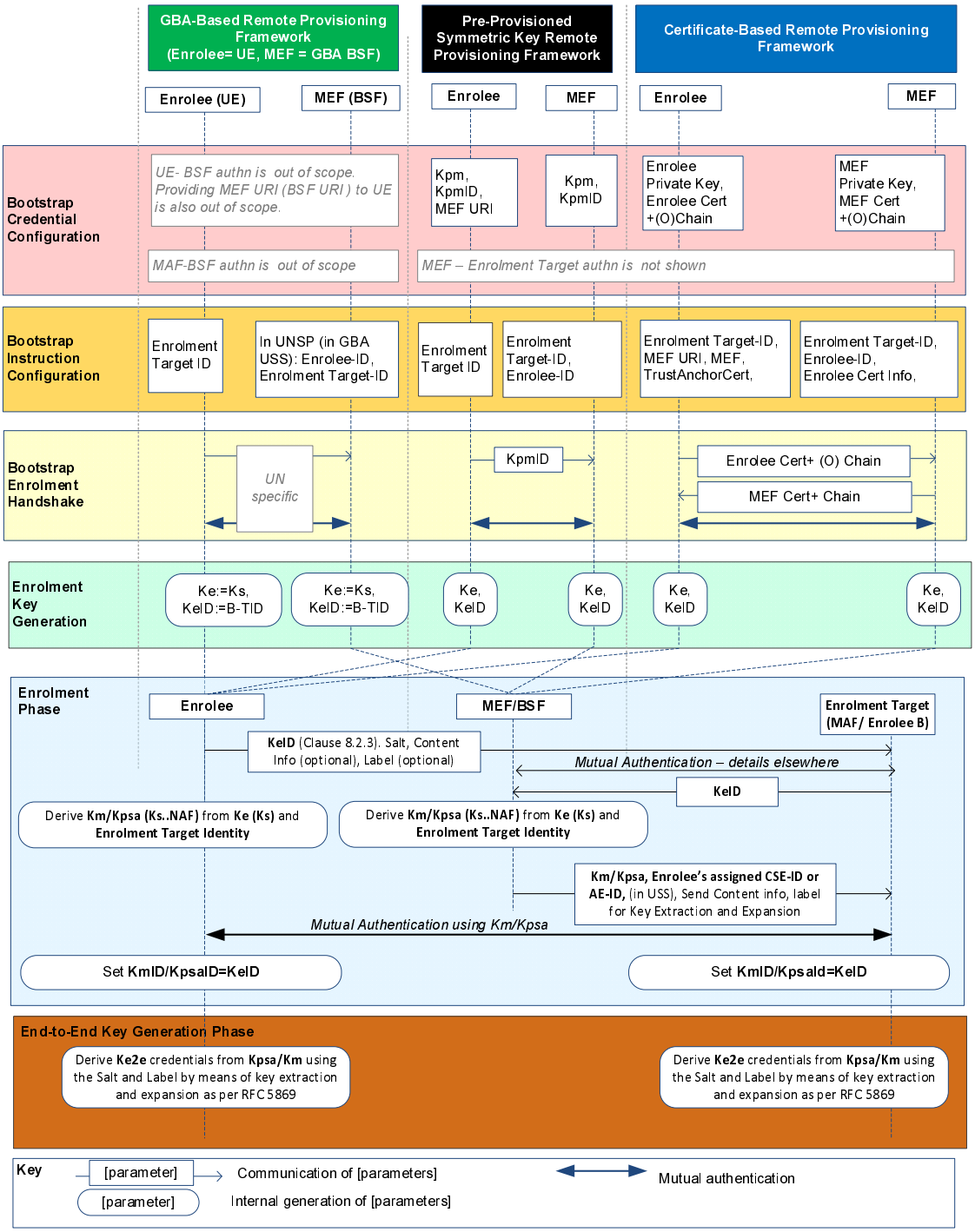


Figure 8.6.2-1: High-level summary of the E2E Remote Security Provisioning Frameworks

Bootstrap Credential Configuration: The Bootstrap Credential Configuration may be based upon the type of Remote Provisioning Framework that is used. When using Symmetric Key Remote Provisioning, the Enrolee, which could be the Source ESF End-Point and the Enrolment targets (Target ESF End-Point) are each pre-provisioned with the Symmetric Enrolee Key (Kpm) and the corresponding Pre-provisioned Symmetric Key Identifier, denoted KpmID. In addition the Source ESF End-Point is provisioned with the Trust Enabling Function address (TEF URI). The mechanism follows the procedures as described in clause 8.3.2.1.

Bootstrap Instruction Configuration: The Source ESF End-Point (Enrolee) and the Trust Enabling Function are configured with the information needed for authorizing the remote provisioning:

- The Source ESF End-Point (Enrolee) is configured with the following arguments to initiate remote provisioning:

  a) The Target ESF End-Point's security profile: The Target ESF End-Point's security profile and the associated security capabilities as described in *<e2ESecurityCapabilities>* resource may be used to identify the types of security protection mechanisms that shall be used for end-to-end security.

  b) The Target ESF End-Point identity: Identifying the Target ESF End-Point for which the Source ESF End-Point is to provision end-to-end security credentials.

  c) The Target ESF End-Point's security profile: The Target ESF End-Point's security profile and the associated security capabilities as described in *<e2ESecurityCapabilities>* resource can be used to identify the types of security protection mechanisms that shall be used for end-to-end security.

  d) The Source ESF End-Point associates these arguments with the Trust Enabling Function (TEF). The Trust Enabling Function can be identified to the Source ESF End-Point using the Pre-Provisioned Symmetric Enrolee Key Identifier (KpmID) and Trust Enabling Function URI.

- M2M Enrolment or Trust Enabling Function is configured with the following arguments to authorize the M2M Enrolment or Trust Enabling Function to remotely provision the Source ESF End-Point for a Target ESF End-Point:

  a) The Target ESF End-Point Identity: Identifying the Target ESF End-Point for which the Source ESF End-Point is to be provisioned.

  b) Source ESF End-Point's assigned CSE-ID or AE-ID (Source ESF End-Point-ID). The Trust Enabling Function is to provide this entity identity for the Source ESF End-Point with the Km or Kpsa to the Target ESF End-Point, when requested by the Target ESF End-Point.

  c) Source ESF End-Point's Security Profile: The security profile of the Source ESF End-Point provides the expected security level described within the *<e2ESecurityCapabilities>* resource (see clause 9.6.3 of ETSI TS 118 101 [1]) associated with the Source ESF End-Point.

  d) Target ESF End-Point's Security Profile: The security profile of the Target ESF End-Point provides the expected security level described within the *<e2ESecurityCapabilities>* resource associated with the Target ESF End-Point.

  e) The Trust Enabling Function shall provide detailed key extraction and expansion parameters that are to be used when deriving the end-to-end credentials from the Km or Kpsa to the Source ESF End-Point and the Target ESF End-Point.

  f) The Trust Enabling Function shall provide the scope and associated security parameters to the Source ESF End-Point and Target ESF End-Point that determines the protocols and the cryptographic algorithms that shall be used for performing end-to-end security.

- **Bootstrap Security Handshake:** The Source ESF End-Point and Trust Enabling Function shall perform a (D)TLS-PSK handshake [15] to establish a secure session. The mechanisms follow the process detailed in clause 8.3.2.

- **End-to-End Key Generation:**

  a) The Enrolment Key (Ke) and RelativeKeID is generated from the (D)TLS session secrets by the Source ESF End-Point and Trust Enabling Function using TLS Key Export (IETF RFC 5705 [18]), as described in clause 10.3.1. Similarly, the Enrolment Key Identifier (KeID) is generated from the RelativeKeID and the Trust Enabling Function's FQDN by the Source ESF End-Point and Trust Enabling Function, as described in clause 10.3.4. The Source ESF End-Point and the Trust Enabling Function store the Ke and the associated KeID.

  b) The end-to-end master key (Ke2e_master) and the E2EKeyId are generated in a similar manner as the Kpsa and the associated KpsaID. If the Source ESF End-Point requests the provisioning of end-to-end keys, then a key extraction based on Kpsa/Km shall be performed.

c) The End-to-End master Key (Ke2e_master) is used to generate specific security protection keys, such as, end-to-end authentication key, end-to-end confidentiality key and other keys depending upon the key extraction and expansion parameters that were provided. The key extraction and expansion are based upon IETF RFC 5869 [48].

NOTE: The End-to-End Key Generation for the Pre-Provisioned Symmetric Enrolee Key Remote Security Provisioning Framework is identical to the End-to-End Key Generation for the Certificate-Based Remote Security Provisioning Framework.

8.6.3 Detailed Description on Source-Generated End-to-End Credentials

This clause describes the Generation and Registration of Symmetric End-to-End Security credentials by a Source ESF End-Point. The end-to-end security credentials that were self-generated by a Source shall be registered with the Trust Enabling Function. Such a mechanism is particularly useful when data (attribute) as well messages targeted for more than one Target are required. In cases, where securing of <*contentInstance*> resource that is consumed by multiple end entities is required, then Source-Generated credentials shall be used.

A Source that generates data consumed by one or more end entities may generate the appropriate credentials so that either a single attribute (e.g. content attribute value of a <*contentInstance*> resource or *customAttribute* of a <*flexContainer*> resource) or a single addressable element within the attribute may be protected for integrity and confidentiality by means of ESData/ESPrim. In the case of dynamic authorization, all or part of a single primitive parameter value (e.g. a signed, self-contained access token communicated in a request primitive to obtain dynamic authorization) may also be protected using ESData/ESPrim. The entity that generated the ESData/ESPrim then registers the credentials with a Trust Enabling Function.

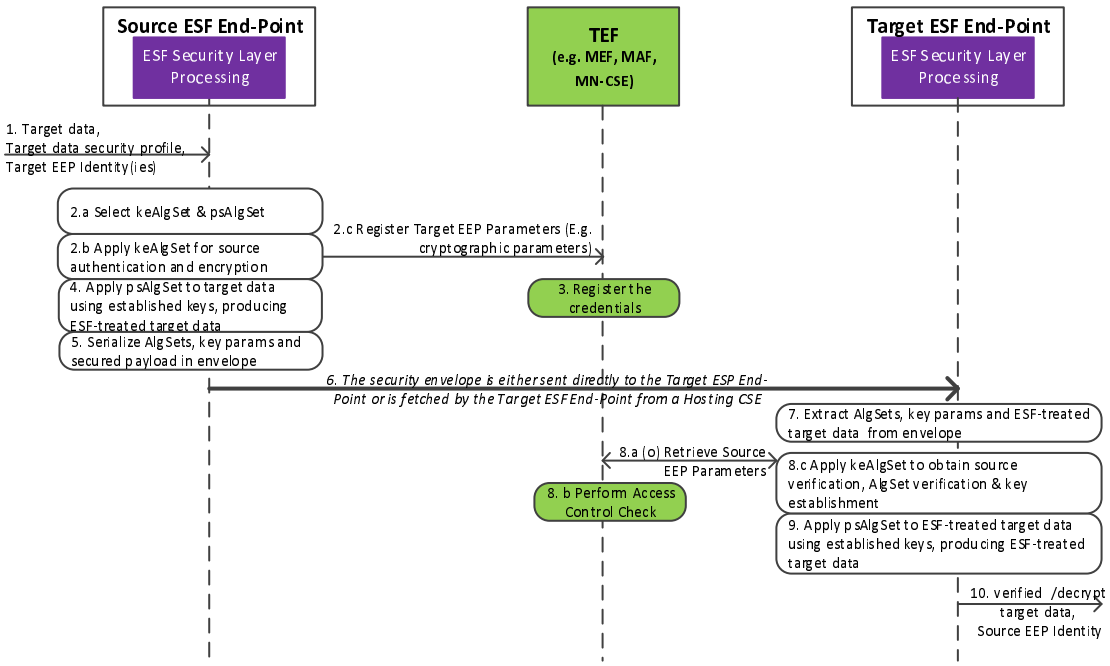


Figure 8.6.3-1: High-level summary of using a TEF for distribution of source-generated credentials

Bootstrap Credential Configuration: It is assumed that the Source ESF End-Point is provisioned with the Ke/KeID that was generated as part of the Remote Provisioning Framework with a Trust Enabling Function (e.g. M2M Enrolment Function) as described in clause 8.2. The Source ESF End-Point may be provisioned with the Km/KmID that was generated as part of the Remote Provisioning Framework with a Trust Enabling Function as described in clause 8.3. The Target ESF End-Point may be provisioned with the Ke/KeID if the Trust Enabling Function is an M2M Enrolment Function, or with the Km/KmID if the security association was established with an M2M Authentication Function.

Bootstrap Instruction Configuration: The Source ESF End-Point as well as the Target ESF End-Points are provisioned with those Trust Enabling Function's URI that support end-to-end security credential provisioning and registration.

- The Source ESF End-Point is configured with the following arguments to initiate remote provisioning:

  a) The identity of the Target ESF End-Point that has to be provided with the ESData/ESPrim and associated End-to-End security credentials.

  b) The security requirement associated with the Data (attributes): This is pre-configured and provided by the application. Based on the security protection mechanisms, appropriate security technologies shall be used for protection.

  c) Pre-configured with a table listing the security requirement and how that security requirement can be achieved using the appropriate security protection mechanisms (e.g. security protocols, algorithms for integrity, confidentiality, key generation).

  d) The Target ESF End-Point Security Profile (optional): The Target ESF End-Point's security profile and the associated security capabilities of the Target ESF End-Point as described in *<e2ESecurityCapabilities>* resource can be used to identify the types of security protection mechanisms that shall be used for providing ESData/ESPrim.

- The Trust Enabling Function is configured with the following arguments to register the ESData/ESPrim security credentials from the Source ESF End-Point and to authorize the Trust Enabling Function to provision only a set of authorized Target ESF End-Point(s) with the relevant ESData/ESPrim security cryptographic parameters:

  a) Cryptographic Parameters: A list of end-to-end cryptographic parameters that is identified by a Credential-Id and having associated cryptographic values such as the credential(s), cryptographic algorithm(s), label(s) and random value(s) (e.g. nonce, IV). These parameters are provided by the Source ESF End-Point during the credential registration process. There may be one or more credentials that are associated with one or more security protection mechanisms (e.g. data integrity, data confidentiality). The list may also include scope and usage of the end-to-end security parameters so that the Target ESF End-Point is able to process ESData/ESPrim (e.g. verify the integrity and/or decrypt the ESData/ESPrim).

  b) Identity of Target ESF End-Point: Identity of the Target ESF End-Points that shall be provisioned with the requested credentials identified by a Credential-Id. The authorization may be provided and enforced by means of ACPs.

- The Target ESF End-Point is configured with the following arguments:

  a) ESData/ESPrim: The Target ESF End-Point is either sent the ESData/ESPrim directly from a Source ESF End-Point, or the Target ESF End-Point fetches the ESData/ESPrim from a hosting entity (e.g. Hosting CSE).

  b) Credential-Id: The Target ESF End-Point is provisioned with the Credential-Id, which may be included as part of the ESData/ESPrim.

  c) Cryptographic Parameters: The Cryptographic Parameters are provisioned by the Trust Enabling Function after the Trust Enabling Function verifies the access control policies associated with the request from the Source ESF End-Point.

- **Security Handshake:**

  a) The Source ESF End-Point and the Trust Enabling Function perform a (D)TLS handshake [15] to establish a secure session. The mechanisms follow the process detailed in clause 8.3.2. All communications between the Source ESF End-Point and the Trust Enabling Function are secured by means of the established (D)TLS connection.

  b) The Target ESF End-Point and the Trust Enabling Function performs a (D)TLS handshake [15] to establish a secure session. The mechanisms follow the process detailed in clause 8.3.2. All communications between the Source ESF End-Point and the Trust Enabling Function are secured by means of the established (D)TLS connection.

- **End-to-End Key Generation:**

  a) The Source ESF End-Point generates credentials that may be based upon:

  - Credentials that have been generated using the Enrolment Key, Ke/KeID that has been generated using the Bootstrapped Remote Credential Provisioning Process.
  - Credentials generated in a random manner by the Source ESF End-Point and registered with the Trust Enabling Function.

8.7 End-to-End Certificate-based Key Establishment (ESCertKE)

8.7.1 Purpose of ESCertKE

End-to-End Certificate-based Key Establishment (ESCertKE) provides an interoperable framework for two end-points to use certificates for establishing a secret symmetric key called *pairwiseE2EKey* from which symmetric keys are derived for use in other end-to-end security frameworks such as End-to-End Security of Data (ESData) or End-to-End Security of Primitives (ESPrim).

Applicable use cases and requirements are discussed in ETSI TR 118 512 [i.16].

The present document specifies the ESCertKE messages and associated processing for ESCertKE. The transport of ESCertKE messages is specified in ETSI TS 118 101 [1].

8.7.2 ESCertKE Architecture

8.7.2.1 ESCertKE Reference Model

The entities in the ESCertKE reference model are the *ESCertKE Initiating End-Point* which initiates the procedure and the single *ESCertKE Terminating End-Point* with which the ESCertKE Initiating End-Point intends to establish a pairwiseE2EKey.

NOTE: Within the scope of clause 8.7, terms including the word "ESCertKE" can be shortened by removing "ESCertKE" to facilitate readability. For example, "ESCertKE Initiating End-Point" is often shortened to "Initiating End-Point".

The *ESCertKE Procedure* consists of the Initiating End-Point and Terminating End-Point exchanging a sequence of *ESCertKE Messages* and apply processing based on those ESCertKE Messages. If the ESCertKE Procedure is successful, then the Initiating End-Point and Terminating End-Point export a pairwiseE2EKey based on the parameters exchanged in the ESCertKE Messages.

There is no inherent restriction on which entities may be an Initiating End-point; these end-points may be entities inside a oneM2M system (that is, AEs and CSEs) or entities outside of a oneM2M system (for example, entities which are part of a system that interworks with oneM2M).

The only restriction on entities which may be Terminating End-Points is that the Terminating End-Point shall be able to receive the unsolicited ESCertKE Message initiating the ESCertKE Procedure. Since ETSI TS 118 101 [1] specifies the transport of ESCertKE messages, ETSI TS 118 101 also specifies which entities may be Terminating End-Points.

8.7.2.2 ESCertKE Procedure Message Flow

The ESCertKE Messages shall be transported as specified in ETSI TS 118 101 [1]; for example, the *<e2EKeyCSE>* resource may be used.

The ESCertKE Messages shall contain the TLS v1.2 [5] messages defined in table 8.7.2.2-1 "ESCertKE Message definitions".

The ESCertKE Procedure message flow is shown in figure 8.7.2.2-1, and described in the following text.

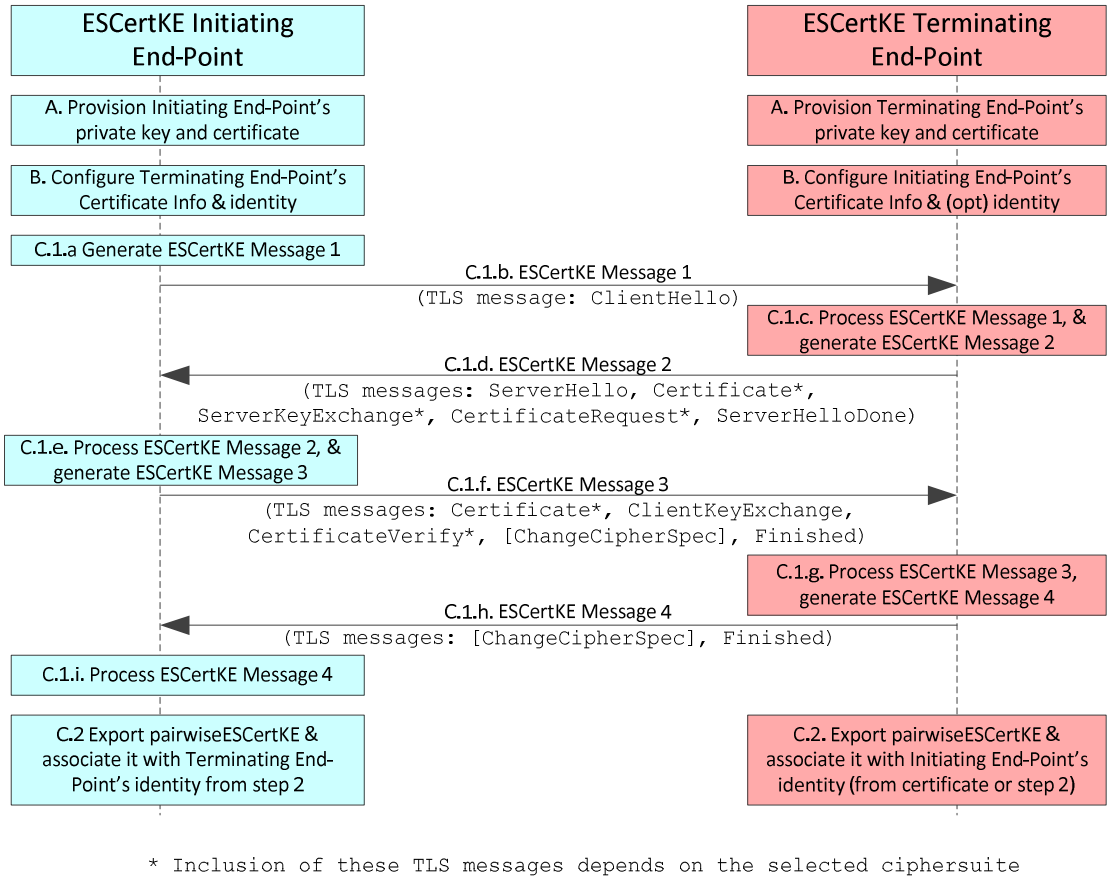


Figure 8.7.2.2-1: ESCertKE Procedure message flow

A. **Provisioning Certificates:** The ESCertKE endpoints shall be provisioned with private key and certificates described in clause 8.1.2.3. The certificates of the Initiating End-Point and terminating End-Points shall conform to clause 10.1.

B. **Triggering:** The Initiating End-Point and Terminating End-Point shall be configured with the information needed for the authentication and identification of the Terminating End-Point and Initiating End-Point respectively:

The Initiating End-Point is commanded to initiate the ESCertKE Procedure, and the command shall include the following arguments:

- The Terminating End-Point's certificate information: as described in clause 8.1.2.4.
- The Terminating End-Point's identity. This identity is used for:
  - Determining where ESCertKE Message 1 is sent; and
  - Associating with the established pairwiseE2EKey.

The Terminating End-Point shall be configured with the following arguments describing Initiating Entity authorized to perform the ESCertKE Procedure:

- The Initiating End-Point's certificate information: as described in clause 8.1.2.4:
  - In the case where the Initiating End-Point's certificate is a raw public key certificate, the Terminating End-Point shall also be configured with an identity to associate with the established pairwiseE2EKey.

The End-Points may be configured in any order.

C. Establishing *pairwiseE2EKey*:

C.1 The Initiating End-Point and Terminating End-Point exchange the sequence of four ESCertKE Messages. The ESCertKE Messages shall be generated and processed according to the handshake protocol of TLS v1.2 [5]. The TLS ciphersuites used for the ESCertKE Procedure shall conform to clause 10.2.3:

C.1.a The Initiating End-Point shall generate ESCertKE Message 1.

C.1.b The Initiating End-Point shall send ESCertKE Message 1 to the Terminating End-Point identified in step 2.

C.1.c The Terminating End-Point shall process ESCertKE Message 1, and generate ESCertKE Message 2.

C.1.d The Terminating End-Point shall send ESCertKE Message 2 to the Initiating End-Point.

C.1.e The Initiating End-Point shall process ESCertKE Message 2, and generate ESCertKE Message 3.

C.1.f The Initiating End-Point shall send ESCertKE Message 3 to the Terminating End-Point.

C.1.g The Terminating End-Point shall process ESCertKE Message 3, and generate ESCertKE Message 4.

C.1.h The Terminating End-Point shall send ESCertKE Message 4 to the Initiating End-Point.

C.1.i The Initiating End-Point shall process ESCertKE Message 4.

C.2 If the TLS handshake protocol is successful, then the Initiating and Terminating End-Points shall export and cache the pairwiseE2EKey using TLS Exporter specification (IETF RFC 5705 [18]) as described in clause 10.3.1.

Table 8.7.2.2-1: ESCertKE Message definitions

ESCertKE Message	Sending End-Point	Possible TLS v1.2 Messages (success case) [5]	Informative Description (normative description is in TLS v1.2 specification [5])
1	Initiating	ClientHello	List of allowed ciphersuites, random value, and indicator to export pairwiseE2EKey.
2	Terminating	ServerHello	Selected ciphersuite, random value, indicator to export pairwiseE2EKey.
		Certificate*	Terminating End-Point's certificate (and optionally certificate chain).
		ServerKeyExchange*	Key exchange parameters generated by the Terminating End-Point. The content of this parameter is dependent on selected ciphersuite.
		CertificateRequest*	Instructs the Initiating End-Point to authenticate itself with a certificate.
		ServerHelloDone	Indicates the end of the message.
3	Initiating	Certificate*	Initiating End-Point's certificate (and optionally certificate chain).
		ClientKeyExchange*	Key exchange parameters generated by the Initiating End-Point. The content of this parameter is dependent on selected ciphersuite.
		CertificateVerify	Provides explicit verification of an Initiating End-Point's certificate.
		[ChangeCipherSpec]	Notifies the Receiving End-Point that subsequent records will be protected under the newly negotiated CipherSpec and keys.
		Finished	MIC on all preceding parameters exchanged in the procedure. The MIC is generated using session secrets established using the preceding parameters.
4	Terminating	[ChangeCipherSpec]	See above.
		Finished	MIC on all preceding parameters exchanged in the procedure. The MIC is generated using session secrets.
NOTE: The inclusion of the TLS messages marked with "*" is dependent on the chosen ciphersuite.			

8.8 MAF Security Framework Details

8.8.1 Introduction to the MAF Security Framework Details

Clause 8.8 describes the common details and procedures used in the MAF-based Security Frameworks; in the present document these frameworks include:

- The MAF-Based Security Association Establishment Framework (SAEF).
- The MAF-Based End-to-End Security of Primitives (ESPrim) Framework.
- The MAF-based End-to-End Security of Data (ESData) Framework.

These frameworks use a MAF to provide authentication and distribution of symmetric key for use by a Source End-Point initiating establishing the symmetric key, and one or more Target End-Points. Table 8.8.1-1 MAF Clients can retrieve the output symmetric key from the MAF. The MAF provides its services on behalf of *administrating stakeholders* such as M2M SPs or third party M2M Trust Enablers (MTE). An administrating stakeholder authorizes the MAF to provide services to MAF clients, and oversees authorizing the distribution of symmetric keys. Table 8.8.1-1 describes the mapping of Source MAF Client and Target MAF Client to roles in the specific MAF-Based Frameworks, and the allowed number of Target MAF Clients.

Table 8.8.1-1: Mapping to specific MAF-based Security Frameworks

MAF-Based Security Framework	Source MAF Client	Target MAF Client	Number of Target MAF Clients	Output Symmetric Key
Security Association Establishment Framework (SAEF)	Entity A	Entity B	1	M2M Secure Connection Key (Kc)
End-to-End Security of Primitives (ESPrim)	Originator	Receiver	1	pairwiseESPrimKey
End-to-End Security of Data (ESData)	Source ESData End-Point	Target ESData End-Point	1..n	ESData Key

This clause 8.8 specifies *MAF Procedures* between the MAF Clients and associated messages. The operation and management of the MAF, beyond the details provided for the MAF Procedures, are not specified in the present document.

The general sequence for using the MAF procedures is shown in figure 8.8.1-1 and described as follows:

1. Each MAF Client shall separately establish credentials for mutual authentication with the MAF as described in **MAF Client Credential Configuration** (clause 8.8.3.1).

2. Each MAF client shall be separately configured to register on the MAF with a specific administrating stakeholder. **MAF Client Registration Configuration** (clause 8.8.3.2) provides the necessary parameters.

3. Each MAF Client shall perform a **MAF Client Registration procedure** with the MAF. This provides confirmation that the MAF Client is willing to use the services of the MAF, under the authorization of the administrating stakeholder. The MAF client shall register separately for each administrating stakeholder, even when registering via a single MAF. If the MAF Client is remotely provisioned for mutual authentication with the MAF, then the MAF shall provide the MAF Client with the KmID to be used for subsequently authentication with the MAF.

   At a later time independent of this sequence of events, the **MAF Client Registration Update procedure** may be performed to confirm that the MAF Client is willing to use the services of the MAF and / or establish a new Km and KmID, and the **MAF Client De- Registration procedure** may be performed to signal that the MAF Client is ceasing to use the services of the MAF.

4. The Source MAF client shall be configured to establish secure communication using a security feature (SAEF, ESPrim or ESData) with symmetric keys established via the MAF. The details of this configuration is specific to the security feature being invoked, but shall include the **MAF Key Registration Configuration** (clause 8.8.3.3).

5. The Source MAF Client shall perform a **MAF Key Registration procedure** to establish a symmetric key and corresponding identifier. The Source MAF Client shall also provide the Security Usage Identifier (SUID) limiting the scope of the credential by identifying the security feature (SAEF, ESPrim or ESData). This procedure shall include the **MAF Handshake procedure** for mutual authentication of the Source MAF Client and MAF.

   At a later time independent of this sequence of events, the **MAF Key Registration Update procedure** may be performed to update the expiration of the registered key or update the list of Target MAF Clients, and the **MAF Key De-Registration procedure** may be performed to delete the key registration from the MAF.

6. The Source MAF Client shall provide, to the Target MAF Client(s), the symmetric key identifier established in the MAF Key Registration procedure. The details of this step depend on the security feature as identified by the SUID.

7. The Target MAF Client shall perform the **MAF Key Retrieval procedure**, to retrieve the symmetric key and corresponding information. This procedure shall include the **MAF Handshake** procedure for mutual authentication of the Target MAF Client and MAF.

8. The symmetric key shall be used in the security protocol between the Source MAF Client and Target MAF Client. If the security protocol requires a single symmetric key, then the first half of the distributed symmetric key shall be used. If the security protocol requires two symmetric keys (for example, an encryption key and a separate integrity key), then the two halves of the distributed symmetric key shall be used as the two security protocol symmetric keys. The details of this step depend on the security feature.

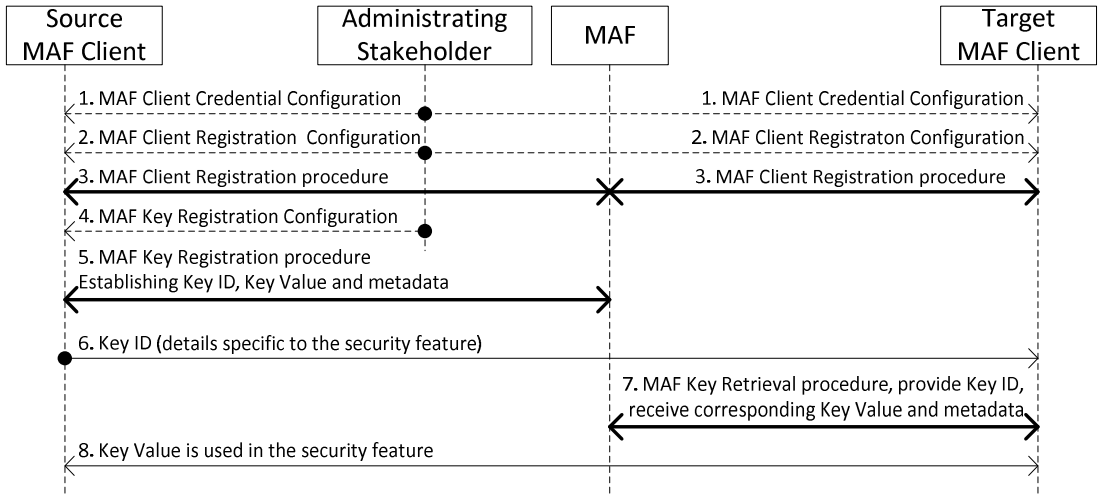


Figure 8.8.1-1: The sequence of events when using the MAF Security Framework as part of a security feature

Clause 8.8 is organized as follows. Clause 8.8.2 describes the processing and information flows of the MAF Procedures. Clause 8.8.3 describes the information in the MAF Client Credential Configuration, MAF Client Registration Configuration and MAF Key Registration Configuration.

8.8.2 MAF Security Framework Processing and Information Flows

8.8.2.1 Introduction

Clause 8.8.2 specifies the processing and information flows of the MAF procedures.

8.8.2.2 MAF Handshake Procedure

Purpose: A MAF Handshake procedure establishes a mutually authenticated TLS or DTLS session for protecting the communication between an MAF Client and MAF. In the case of the MAF Key Registration procedure, the TLS or DTLS session may be used by the Source MAF Client and MAF to establish the Key Value.

Pre-Conditions: One of the following conditions shall hold:

- The MAF Client and MAF have been provisioned with certificates as described in the MAF Client Credential Configuration details in clause 8.8.3.1, and configured with CA certificates for validating certificates as described in the MAF Client Registration Configuration details in clause 8.8.3.2.
- The MAF Client and MAF have established a symmetric Master Credential (Km) with corresponding Master Credential Identifier (KmID). The Km and KmID may be pre-provisioned, or Km may be established using Remote Security Provisioning Framework with KmID established using the MAF Client Registration procedure.

  NOTE: In the case of establishing Km via remote provisioning, MAF Handshake cannot be performed during MAF Client Registration because (a) the MAF does not know Km prior to MAF Client Registration and (b) KmID has not been assigned prior to MAF Client Registration.

Procedure description:

- If the MAF Client and MAF have established a symmetric Master Credential (Km) with corresponding Master Credential Identifier (KmID), then the MAF Client and MAF shall establish the TLS or DTLS session using the TLS-PSK handshake according to clause 10.2.2, with the following details:
  - The "psk_identity" parameter [15] shall be set to the value of the Master Credential Identifier (KmID).
  - The "psk" parameter [15] shall be set to the value of the Master Credential (Km).
- If the MAF Client and MAF are to authenticate using certificates, then the MAF Client and MAF shall establish the TLS or DTLS session using the certificate-based TLS handshake according to clause 10.2.2, with the following details:
  - The TLS server certificate shall be the MAF's certificate. The MAF Client shall verify the MAF's certificate against the set of provisioned MAF certificate trust anchors as described in clause 8.1.2.5.
  - The TLS client certificate shall be the 'MAF Client's certificate. The MAF shall verify the 'MAF Client's certificate against the provisioned MAF Client Certificate Information as described in clause 8.1.2.5.

8.8.2.3 MAF Client Registration Procedure

Purpose: The MAF Client registers with the MAF to confirm that it is willing to use the services of the MAF, under the authorization of the administrating stakeholder. If remote provisioning is used to establish a symmetric key between an MAF Client and the MAF, then the MAF Client triggers the MAF (in the TLS handshake) to retrieve Km from the MEF, and the MAF provides the MAF Client with the Master Credential Id (KmID) to use in subsequent MAF Handshake Procedures.

  NOTE: The MAF Client Registration procedure is equivalent to CSE or AE registration, but in this case the MAF Client is "registering" to the MAF, and not the registrar CSE.

Pre-Conditions: The MAF Client, MAF, and (where applicable) MEF have been provisioned with the parameters described in clause 8.8.3.1 and 8.8.3.2.

Procedure description:

1. The MAF Client shall establish a TLS (or DTLS) connection with the MAF.
   - If remote provisioning is used, then steps (b) onwards in the "Use of Provisioned Credential" in clause 8.3.2.1 shall be performed by the MAF Client (assuming the role of Enrolee), MEF and MAF (assuming the role of Registration Target). The SUID of the remotely-provisioned key shall be '21' "A symmetric key, provisioned via a Remote Security Provisioning Framework (RSPF), and intended to be shared with a MAF" as specified in ETSI TS 118 104 [4]. The MAF retrieves Km from the MEF as part of this process.
   - Otherwise, the MAF Client and MAF shall perform the MAF Handshake Procedures (clause 8.8.2.2).

   This provides the MAF with an authenticated identity for the MAF Client.

2. The MAF Client shall send a MAF Client Registration request including the information shown in table 8.8.2.3-1.

Table 8.8.2.3-1: MAF Client Registration Request message information

Parameter	Description	Multiplicity
MAF-FQDN	FQDN of the MAF, from MAF Instruction Configuration	1
expirationTime	Proposed time when the registration shall expire.	1
labels	Labels to aid discovery the record of the MAF Client's registration	0..1
adminFQDN	FQDN of the administrating stakeholder, provided in the MAF Client Registration Configuration	1

3. Upon receiving the request, the MAF shall process the request. If error cases are encountered, then the MAF shall send an error response. The MAF may assign different values for parameters received from the MAF Client, based on instruction from the administrating stakeholder. If the request is processed successfully, then the MAF shall compose a MAF Client Registration response request including the information shown in table 8.8.2.3-2.

Table 8.8.2.3-2: MAF Client Registration Response message information

Parameter	Description	Multiplicity
mafClientRegID	An identifier for the new MAF Client Registration record	1
labels	Labels to aid discovery of the MAF Client Registration record	0..1
expirationTime	Time when the MAF Client Registration record shall expire	1
MAF Client ID	Identifier of the MAF Client	1
adminFQDN	FQDN of the administrating stakeholder	1
assignedSymmKeyID	MAF-Assigned Master Credential ID (KmID), in cases where the Km is remotely provisioned	0..1

The MAF shall send the response to the MAF Client.

4. The MAF Client and MAF shall store the parameters. If assignedSymmKeyID was included, then the MAF Client shall use this as Master Credential ID (KmID) hereafter when establishing TLS (or DTLS) sessions with the MAF.

8.8.2.4 MAF Client Configuration Retrieval Procedure

Purpose: This procedure enables a MAF Client to retrieve MAF Client Configurations provided by the administrating stakeholder to the MAF.

Pre-Conditions:

- The MAF Client has previously performed the MAF Client Registration procedure to create the MAF Client Registration record.
- The MAF Client Registration record is not expired.

Procedure Description. The procedure comprises the following steps:

1. The MAF Client shall establish a TLS (or DTLS) connection with the MAF as described in step 1 of clause 8.8.2.3.
2. The MAF Client shall send a MAF Client Configuration Retrieval request including the information shown in table 8.8.2.4-1.

Table 8.8.2.4-1: MAF Client Configuration Retrieval Request message information

Parameter	Description	Multiplicity
MAF-FQDN	FQDN of the MAF, from MAF Instruction Configuration	1
mafClientRegID	Identifier for the MAF Client registration record being updated	1

3. Upon receiving the request, the MAF shall process the request. If error cases are encountered, including if there is no MAF Client Configuration currently associated with the identified MAF Client registration record, then the MAF shall send an error response. If the request is processed successfully, then the MAF shall attempt to retrieve the MAF Client Configuration currently associated with the identified MAF Client registration record.

4. The MAF shall compose a MAF Client Configuration Retrieval response a containing the following parameters.

Table 8.8.2.4-2: MAF Client Configuration Retrieval Response message information

Parameter	Description	Multiplicity
mafClientCfg	MAF Client Configuration currently associated with the identified MAF Client registration record	1

The MAF shall send the response to the MAF Client.

5. The MAF Client shall apply the MAF Client Configuration.

8.8.2.5 MAF Client Registration Update Procedure

Purpose: This procedure enables a MAF Client to update the MAF Client registration by any combination of extending the *expirationTime* of the MAF Client Registration record, updating the *labels* or establish a new Km and KmID.

Pre-Conditions:

- The MAF Client has previously performed the MAF Client Registration procedure to create the MAF Client Registration record.
- The MAF Client Registration record is not expired.

Procedure Description. The procedure comprises the following steps:

1. The MAF Client shall establish a TLS (or DTLS) connection with the MAF as described in step 1 of clause 8.8.2.3.

2. The MAF Client shall send a MAF Client Registration Update request including the information shown in table 8.8.2.5-1.

Table 8.8.2.5-1: MAF Client Registration Update Request message information

Parameter	Description	Multiplicity
MAF-FQDN	FQDN of the MAF, from MAF Instruction Configuration	1
mafClientRegID	Identifier for the MAF Client registration record being updated	1
expirationTime	Proposed time when the MAF Client registration record shall expire.	0..1
labels	Labels to aid discovery of the MAF Client registration record	0..1
NOTE: At least one of expirationTime and labels shall be included.		

3. Upon receiving the request, the MAF shall process the request. If error cases are encountered, then the MAF shall send an error response. If the request is processed successfully, then the MAF shall update the MAF Client Registration record with the proposed values if authorized by the administrating stakeholder. The MAF may assign different values for parameters received from the MAF Client, based on instruction from the administrating stakeholder.

4. The MAF shall compose a MAF Client Registration Update response a containing the following parameters.

Table 8.8.2.5-2: MAF Client Registration Update Response message information

Parameter	Description	Multiplicity
expirationTime	Updated time when the MAF Client Registration record shall expire.	0..1
labels	Updated labels to aid discovery of the MAF Client Registration record.	0..1
assignedSymmKeyID	MAF-Assigned Master Credential ID (KmID), in cases where a new Km is remotely provisioned.	0..1
NOTE: The response only includes *expirationTime* and/or *labels* if those parameters were present in the corresponding request.		

The MAF shall send the response to the MAF Client.

5. The MAF Client and MAF shall store the parameters. If *assignedSymmKeyID* was included, then the MAF Client shall use this as Master Credential ID hereafter when establishing TLS (or DTLS) sessions with the MAF.

8.8.2.6 MAF Client De-Registration Procedure

Purpose: This procedure enables a MAF Client to end its registration with the MAF.

Pre-Conditions:

- The MAF Client has previously performed the MAF Client Registration procedure to create the MAF Client Registration record.
- The MAF Client Registration record is not expired.

Procedure Description. The procedure comprises the following steps:

1. The MAF Client shall establish a TLS (or DTLS) connection with the MAF as described in step 1 of clause 8.8.2.3.

2. The MAF Client shall send a MAF Client De-Registration request including the information shown in table 8.8.2.6-1.

Table 8.8.2.6-1: MAF Client De-Registration Request message information

Parameter	Description	Multiplicity
MAF-FQDN	FQDN of the MAF, from MAF Instruction Configuration	1
mafClientRegID	Identifier for the MAF Client Registration record being ended	1

3. Upon receiving the request, the MAF shall process the request. If error cases are encountered, then the MAF shall send an error response. If the request is processed successfully, then the MAF shall delete the information associated with the identified MAF Client Registration record.

4. The MAF shall compose a MAF Client Registration Update response indicating the success of the operation. The MAF shall send the response to the MAF Client.

8.8.2.7 MAF Key Registration Procedure

Purpose: This procedure enables a Source MAF Client to establish a symmetric key with the MAF which can be retrieved for use by one or more Target MAF Clients.

This procedure is performed between the Source MAF Client and the MAF.

Pre-Conditions:

- The Source MAF Client is provided with (or has otherwise determined) the information in the MAF Key Registration Configuration (clause 8.8.3.3).
- The Source MAF Client has performed the MAF Client Registration Procedure (clause 8.8.2.3) with the MAF for the administrating stakeholder identified in the MAF Key Registration Configuration.

Procedure Description. The procedure comprises the following steps:

1. The Source MAF Client shall establish a TLS or DTLS session with the MAF using the MAF Handshake procedure, described in clause 8.8.2.2. A by-product of the MAF Handshake procedure is that the MAF establishes an authenticated identity for the Source MAF Client.
2. The Source MAF Client selects the value of the M2M Secure Connection Key (Kc) to be distributed by the MAF. The value shall be one of the following:
   - The Source MAF Client generates the output symmetric key value from the (D)TLS session secrets using TLS Key Export (IETF RFC 5705 [18]), as described in clause 10.3.1.
   - The output symmetric key value is self-generated by the Source MAF Client, independently of the (D)TLS session secrets.
3. The Source MAF Client shall compose a list of Target MAF Clients to whom the MAF is authorized to provide the output symmetric key value:
   - In the case of MAF-Based SAEF or MAF-Based ESPrim: The list shall contain exactly one Absolute AE-ID or Absolute CSE-ID.
   - In the case of MAF-Based ESData: The list shall contain any non-zero number of Absolute AE-ID or Absolute CSE-IDs.

NOTE 1: How the Source MAF Client selects the list of Target MAF Clients is application dependent.

4. The Source MAF Client shall send a MAF Key Registration request, including the information shown in table 8.8.2.7-1. If the Key Value is not present in the request, the MAF client shall generate Key Value from the (D)TLS session using the TLS Key Export (IETF RFC 5705 [18]), as described in clause 10.3.1.

Table 8.8.2.7-1: MAF Key Registration Request message information

Parameter	Description	Multiplicity
MAF-FQDN	FQDN of the MAF, from MAF Instruction Configuration	1
expirationTime	Proposed time when the Key Registration shall expire	1
labels	Labels to aid discovery of the Key Registration	0..1
adminFQDN	Identifier for the administrating stakeholder	1
SUID	The Security Usage Identifier limiting the security feature in which the symmetric key may be used.	1
targetIDs	(Optional) list of identifiers for the initial set of Target MAF Clients authorized to retrieve the symmetric key	0..1
Key Value	(Optional) If present, this parameter contains an output symmetric key value which is self-generated by the Source MAF Client. If this parameter is not present, then the Source MAF Client and MAF will generate the output symmetric key value using TLS Exporter	0..1

5. The MAF shall process the request. If error cases are encountered, then the MAF shall send an error response. If the request is processed successfully, then the MAF shall authorize establishing a Key Value, based on the authenticated identity for the Source MAF Client.

NOTE 2: The present document provides no details for the authorization of this request.

6. If the request included a value in the Key Value parameter, then the MAF shall store this value. Otherwise, the MAF shall generate Key Value from the (D)TLS session using TLS Key Export (IETF RFC 5705 [18]), as described in clause 10.3.1.

7. The MAF shall initialize the list of authorized Target MAF Clients (those MAF Clients which may retrieve this credential) to the list provided in the request:

   - In the case of MAF-Based ESData: This list may be further updated by administrating stakeholders during or after the MAF Key Registration procedure.

NOTE 3: The present document does not provide any details about administrating stakeholders updating the list of authorized Target MAF Clients on the MAF. The MAF could provide its own logic and interface allowing administrating stakeholders to manage this list.

8. The MAF shall select a previously-unused value of RelativeKeyID.

9. The MAF may assign different values for parameters received from the MAF Client, based on instruction from the administrating stakeholder.

10. The MAF shall send a response, to the Source MAF Client, including the information shown in table 8.8.2.7-2.

Table 8.8.2.7-2: MAF Key Registration response message information

Parameter	Description	Multiplicity
RelativeKeyID	The relative part of the Key Identifier associated with the Key Registration.	1
expirationTime	Time when the Key Registration shall expire.	1
Source MAF Client ID	Identifier of the Source MAF Client.	1
labels	Labels to aid discovery of the Key Registration.	0..1
adminFQDN	Identifier for the administrating stakeholder.	1
SUID	The Security Usage Identifier limiting the security feature in which the symmetric key may be used.	1
targetIDs	List of identifiers for the initial set of Target MAF Clients authorized to retrieve the symmetric key. This list may have been modified from the list provided by the MAF Client, or created by the MAF (if the MAF Client did not provide a list).	1

11. The Source MAF Client and MAF shall store the output symmetric key value and corresponding Key Identifier:

   - The Key Identifier is generated from the RelativeKeyID and the M2M Authentication Function's FQDN by the Source MAF Client and MAF, as described in clause 10.3.5.

8.8.2.8 MAF Key Retrieval Procedure

Purpose: This procedure enables a Target MAF Client to retrieve the Key Value from a MAF corresponding to a RelativeKeyID received by the Target MAF Client.

Pre-Conditions:

- The Target MAF Client has performed the MAF Client Credential Configuration (clause 8.8.2.1) with the MAF, including configuration of the MAF Key Retrieval URI.

- The Source MAF Client has performed the MAF Key Registration Procedure (clause 8.8.2.2) with the MAF, resulting in a registered Key Value and assigned RelativeKeyID for a specific administrating stakeholder and Security Usage Identifier (SUID).

- The Target MAF Client received a Key Identifier from the Initiating-MAF Client in a security feature with the SUID which the Source MAF Client provided to the MAF during the MAF Key Registration Procedure (clause 8.8.2.7). The Key Identifier shall be composed of the FQDN of the MAF and the RelativeKeyID assigned to the registered key.

- The Target MAF Client may expect that it is authorized to obtain the corresponding output symmetric key value.

NOTE: The Target MAF Client does not have to repeat this procedure if the Target MAF Client is already in possession of the corresponding Key Value.

Procedure Description. The procedure comprises the following steps:

1. The Target MAF Client shall establish a TLS or DTLS session with the MAF using the MAF Handshake procedure, described in clause 8.8.2.2. A by-product of the MAF Handshake procedures is that the MAF establishes an authenticated identity for the Target MAF Client.

2. The Target MAF Client shall send a MAF Key Retrieval request to the MAF including the information shown in table 8.8.2.8-1.

Table 8.8.2.8-1: MAF Key Retrieval Request message information

Parameter	Description	Multiplicity
RelativeKeyID	The relative part of the Key Identifier received from the Source MAF Client in a security feature	1

3. The MAF shall process the request. If error cases are encountered, then the MAF shall send an error response. If the request is processed successfully, then the MAF shall identify the key registration using the *RelativeKeyID*.

4. The MAF shall determine if the Target MAF Client is authorized to retrieve the registered key and metadata by comparing the authenticated identifier for Target MAF Client against the list of identifiers for authorized Target MAF Clients. If the Target MAF Client is not authorized, then the MAF shall send, to the Target MAF Client, an error message. Otherwise, the MAF shall proceed to the next step.

5. The MAF shall send a response, to the Target MAF Client, including the information shown in table 8.8.2.8-2.

Table 8.8.2.8-2: MAF Key Retrieval response message information

Parameter	Description	Multiplicity
expirationTime	Time when the Key Registration shall expire	1
Source MAF Client ID	Identifier of the Source MAF Client	1
labels	Labels to aid discovery of the Key Registration	0..1
adminFQDN	Identifier for the administrating stakeholder	1
SUID	The Security Usage Identifier limiting the security feature in which the symmetric key may be used	1
Key Value	The registered value of the output symmetric key	1

6. The Target MAF Client shall associate the parameters with the key identifier.

8.8.2.9 MAF Key Registration Update Procedure

Purpose: This procedure enables a Source MAF Client to update the metadata associated with a registered key.

This procedure is performed between the Source MAF Client and the MAF.

Pre-Conditions:

- The MAF Client has previously performed the MAF Key Registration procedure to create the key registration.
- The key registration is not expired.

Procedure Description. The procedure comprises the following steps:

1. The MAF Client shall establish a TLS (or DTLS) connection with the MAF as described in step 1 of clause 8.8.2.7.

2. The Source MAF Client shall compose a list of Target MAF Clients to whom the MAF is authorized to provide Kc:

   - In the case of MAF-Based SAEF or MAF-Based ESPrim: The list shall contain exactly one Absolute AE-ID or Absolute CSE-ID.

   - In the case of MAF-Based ESData: The list shall contain any non-zero number of Absolute AE-ID or Absolute CSE-IDs.

NOTE: The present document does not provide any details about how the Source MAF Client selects the list of Target MAF Clients.

3. The Source MAF Client shall send a MAF Key Registration Update request, including the updated information shown in table 8.8.2.9-1.

Table 8.8.2.9-1: MAF Key Registration Update Request message information

Parameter	Description	Multiplicity
MAF-FQDN	FQDN of the MAF, from MAF Instruction Configuration	1
RelativeKeyID	The relative part of the Key Identifier associated with the Key Registration	1
expirationTime	Proposed time when the Key Registration shall expire.	0..1
labels	Proposed Labels to aid discovery of the registered key	0..1
targetIDs	(Optional) proposed list of identifiers for the set of Target MAF Clients authorized to retrieve the symmetric key.	0..1
NOTE: At least one of *expirationTime*, labels or *targetIDs* shall be provided.		

4. The MAF shall process the request. If error cases are encountered, then the MAF shall send an appropriate error response. If the request is processed successfully, then the MAF shall update the metadata with the proposed values if authorized by the administrating stakeholder. The MAF may assign different values for parameters received from the MAF Client, based on instruction from the administrating stakeholder.

5. The MAF shall send a response, to the Source MAF Client, including the information shown in table 8.8.2.9-2.

Table 8.8.2.9-2: MAF Key Registration Update response message information

Parameter	Description	Multiplicity
expirationTime	Current time when the key registration shall expire, if changed since the last time the MAF Client was provided with the expiration time.	0..1
labels	Updated list of labels to aid discovery of the Key Registration, if any.	0..1
targetIDs	Current list of identifiers for the initial set of Target MAF Clients authorized to retrieve the symmetric key. This list may have been modified from the list provided by the MAF Client.	0..1
NOTE: The response includes only those parameters that were present in the corresponding request.		

8.8.2.10 MAF Key De-Registration Procedure

Purpose: This procedure enables a Source MAF Client to request the MAF to stop distributing the registered key.

This procedure is performed between the Source MAF Client and the MAF.

Pre-Conditions:

- The MAF Client has previously performed the MAF Key Registration procedure to create the key registration.
- The key registration is not expired.

Procedure Description. The procedure comprises the following steps:

1. The MAF Client shall establish a TLS (or DTLS) connection with the MAF as described in step 1 of clause 8.8.2.7.

2. The MAF Client shall send MAF Key De-Registration request including the information shown in table 8.8.2.10-1.

Table 8.8.2.10-1: MAF Client De-Registration Request message information

Parameter	Description	Multiplicity
MAF-FQDN	FQDN of the MAF, from MAF Instruction Configuration	1
RelativeKeyID	The relative part of the Key Identifier associated with the Key Registration	1

3. Upon receiving the request, the MAF shall process the request. If error cases are encountered, then the MAF shall send an error response. If the request is processed successfully, then the MAF shall delete the information associated with the identified key registration.

4. The MAF shall compose MAF Client De-Registration response indicating the success of the operation. The MEF shall send the response to the MAF Client.

8.8.3 MAF Client Configuration Details

8.8.3.1 MAF Client Credential Configuration Details

The MAF Client and MAF shall be configured with credentials for mutual authentication of the MAF Client and MAF.

The credentials for mutual authentication shall be either pre-provisioned or remotely provisioned thanks to Remote Security Provisioning Frameworks. Either symmetric key credentials or certificate credentials maybe provisioned. Symmetric key credentials may be used for authenticating some MAF Clients and certificate credentials may be used for authenticating other MAF Clients. The selection may be based on the capabilities of the MAF Client.

The details depend on the type of credential (symmetric key or certificates) and, in the case of symmetric keys, the type of provisioning (pre-provisioning or remote provisioning).

1) Details specific to **Pre-Provisioned Symmetric Keys (PPSKs)**: the Master Credential (Km) and corresponding Master Credential Identifier (KmID) shall be provisioned to the MAF Client (assuming the role of Enrolee) and the MAF. The format of KmID is defined in clause 10.6.

2) Details specific to **Remotely-Provisioned Symmetric Keys (RPSKs)**: The MAF Client and an M2M Enrolment Function (MEF) shall be provisioned with credentials for performing a Remote Security Provisioning (RSPF) Framework. The MAF Client shall be authorized to use the services of the MEF. For more details, see clause 8.3.

NOTE 1: In this case, the Master Credential (Km) and Master Credential Identifier (KmID) are established during the MAF Client Registration procedure.

3) Details specific to **Certificates (whether pre-provisioned or remotely provisioned)**: The MAF Client shall be provisioned with an MAF Client certificate with optional certificate chain. The MAF Client certificate shall be a device certificate, Node-ID certificate, AE-ID certificate or CSE-ID certificate.

NOTE 2: The configuration of MAF trust anchor CA certificates is addressed in MAF Client Registration Configuration, and can occur separately from MAF Client Credential Configuration.

The oneM2M Device Configuration specification ETSI TS 118 122 [57] provides a set of <*mgmtObj*> specializations that shall be used for MAF Client Credential Configuration when the MAF Client supports device management (either remotely or via manual input). The present document does not specify how the MAF Client Credential Configuration is represented when the MAF Client does not support device management.

8.8.3.2 MAF Client Registration Configuration Details

Purpose: The MAF Client Registration Configuration describes the information provisioned to a MAF Client to enable it to perform MAF procedures authorized by an administrating stakeholder. The administrating stakeholder arranges for the MAF Client Registration Configuration to be provided to the MAF Client.

Pre-conditions:

- The MAF Client and MAF have been configured with credentials which can be used for mutual authentication: see MAF Client Credential Configuration in clause 8.8.3.1.
- If the MAF Client and MAF will use certificates for mutual authentication, then:
  - The administrating stakeholder (or another stakeholder acting on behalf of the administrating stakeholder) possesses a copy of the MAF Client's Certificate Information as defined in clause 8.1.2.4. The MAF is provided with a copy of the MAF Client's Certificate Information. The present document does not specify how this information is provided to the MAF by the administrating stakeholder (or another stakeholder acting on behalf of the administrating stakeholder).
  - The administrating stakeholder (or another stakeholder acting on behalf of the administrating stakeholder) possesses a copy of the MAF Trust Anchor CA Certificates. The MAF Client is provided with a copy of the MAF Trust Anchor CA Certificates.
- The administrating stakeholder arranges for the MAF to allow the MAF Client to perform MAF Client Registration. This could involve pre-authorization or real-time authorization.

Details:

The MAF Client Registration Configuration (*mafClientRegCfg*) includes the information shown in table 8.8.3.2-1, and has data type sec:clientRegCfg (see clause 12.4.2).

Table 8.8.3.2-1: Information in the MAF Client Registration Configuration

Element name	Multiplicity	Notes
expirationTime	0..1	Time when the configuration expires
labels	0..1	List of labels to enable discovery of the MAF Client registration record
fqdn	1	MAF-FQDN (also known as MAF-ID)
adminFQDN	1	FQDN of the administrating stakeholder
httpPort	0..1	Port number when using HTTP [i.20]
coapPort	0..1	Port number when using CoAP [i.21]
websocketPort	0..1	Port number when using WebSocket [i.19]

8.8.3.3 MAF Key Registration Configuration Details

Purpose: The MAF Key Registration Configuration describes the information provisioned to a MAF Client to enable it to perform MAF procedures authorized by an administrating stakeholder. The administrating stakeholder arranges for the MAF Client Registration Configuration to be provided to the MAF Client.

Pre-conditions:

- The MAF Client has performed the MAF Client Registration procedure with the MAF for the administrating stakeholder.
- The MAF Client has currently-valid credentials for mutual authentication with the MAF.

Details:

The MAF Key Registration Configuration (*mafKeyRegCfg*) includes the information shown in table 8.8.3.3-1, and has data type sec:keyRegCfg (see clause 12.4.3).

Table 8.8.3.3-1: Information in the MAF Key Registration Configuration

Element Name	Multiplicity	Notes
expirationTime	0..1	Expiration time
labels	0..1	List of labels to enable discovery of the key registration
adminFQDN	1	FQDN of the administrating stakeholder
SUID	1	SUID constraining the usage of the Key Value established during the MAF Key Registration procedure.
targetIDs	0..1	List of identifiers for authorized target MAF Clients

9 Security Framework Procedures and Parameters

9.0 Introduction

This clause specifies procedures and parameters of the phases of Security Association Establishment Frameworks (clause 8.2) and Remote Security Provisioning Frameworks (clause 8.3).

9.1 Security Association Establishment Framework Procedures and Parameters

9.1.1 Credential Configuration Parameters

9.1.1.0 Introduction

The following Credential Configuration procedures are described in the present clause:

- Credential Configuration of Entity A and Entity B, see clause 9.1.1.1.
- Credential Configuration of M2M Authentication Functions, see clause 9.1.1.2.

9.1.1.1 Credential Configuration of Entity A and Entity B

Table 9.1.1.1-1 lists the parameters that may be configured to Entity A during the Credential Configuration phase and which are common to all Security Association Establishment Frameworks.

Table 9.1.1.1-1: Parameters that may be configured to Entity A during the Credential Configuration phase and which are common to all Security Association Establishment Frameworks

Parameter common to all Security Association Establishment Frameworks
(If Entity A is a CSE) Entity A's CSE-ID

Table 9.1.1.1-2 lists the parameters configured to a Field-Domain Security Association End-Points in the Credential Configuration phase and which are specific to the Security Association Establishment Framework.

Table 9.1.1.1-2: Parameters configured to a Field Domain Security Association end-point during the Credential Configuration phase and which are specific to a Security Association Establishment Framework

Security Association Establishment Framework		Parameter
Provisioned Symmetric Key		Kpsa
		KpsaID
Certificate Based	Entity authenticates itself using a Raw Public Key Certificate	Entity's Private Key
		Entity's Raw Public Key Certificate
	Entity authenticates itself using a Device Certificate	Entity's Private Key
		Entity's Certificate and Chain
	Entity authenticates itself using a CSE-ID Certificate	Entity's CSE-ID
		Entity's Private Key
		Entity's Certificate and Chain
	Entity authenticates itself using an AE-ID Certificate	Entity's AE-ID
		Entity's Private Key
		Entity's Certificate and Chain
	Entity authenticates itself using a Node-ID Certificate	Entity's Node-ID
		Entity's Private Key
		Entity's Certificate and Chain
MAF-Based	Entity A	MAF Identifier (MAF-ID)
		Master Credential (KmID)
		Master Credential Identifier (KmID)
	Entity B	Entity B and MAF shall be able to establish mutually-authenticated secure communication. The details are not specified in the present document.

The Credential Configuration of Entity A and Entity B for the Provisioned Symmetric Key Security Association Establishment Framework, or the MAF-Based Security Association Establishment Framework is achieved through either:

- Pre-provisioning via mechanisms which are not specified in the present document.
- Remote provisioning via one of the Remote Security Provisioning Frameworks in clause 8.3.

The Credential Configuration of Entity A and Entity B for the Certificate Security Association Establishment Frameworks is performed by pre-provisioning via mechanisms which are not specified in the present document.

9.1.1.2 Credential Configuration of M2M Authentication Functions

Table 9.1.1.2-1 lists the parameters configured to M2M Authentication Functions in the Credential Configuration phase. The M2M Authentication Function's identifier (MAF-ID) is presumed to have been configured prior to the Credential Configuration phase.

Table 9.1.1.2-1: Parameters configured to a M2M Authentication Functions during the Credential Configuration phase

Security Association Establishment Framework		Parameter
MAF-Based	A-to-MAF Authentication	Master Credential (Km)
		Master Credential Identifier (KmID)
	B-to-MAF Authentication	Entity B and MAF shall be able to establish mutually-authenticated secure communication. The details are not specified in the present document.

The Credential Configuration of M2M Authentication Framework shall be achieved through either:

- Business logic of the Stakeholder operating the M2M Authentication Function, and the details are not described in the present document.
- Remote provisioning via one of the Remote Security Provisioning Frameworks in clause 8.3.

9.1.2 Association Configuration Procedures and Parameters

9.1.2.0 Introduction

The following Association Configuration procedures are described in this clause:

- Association Configuration of Entity A, see clause 9.1.2.1.1.
- Association Configuration of Entity B, see clause 9.1.2.1.2.
- Association Configuration of M2M Authentication Functions, see clause 9.1.2.2.

9.1.2.1 Association Configuration of Entity A and Entity B

9.1.2.1.1 Association Configuration of Entity A

Table 9.1.2.1.1-1 lists the parameters configured to Entity A in the Association Configuration phase and which are common to all Security Association Establishment Frameworks.

Table 9.1.2.1.1-1: Parameters configured to Entity A during the Association Configuration phase and which are common to all Security Association Establishment Frameworks

Parameter common to all Security Association Establishment Frameworks
Entity B's CSE-ID

Table 9.1.2.1.1-2 lists the parameters configured to Entity A in the Association Configuration phase which are specific to the Security Association Establishment Framework.

Table 9.1.2.1.1-2: Parameters configured to Entity A during the Association Configuration phase which are specific to a Security Association Establishment Framework

Security Association Establishment Framework		Parameters specific to the Security Association Establishment Frameworks
Provisioned Symmetric Key		None
Certificate Based	Entity B is authenticated using Raw Public Key Certificate	Entity B's Public key identifier
	Entity B is authenticated using Device Certificate	Entity B's globally unique hardware instance identifier
		Entity B's trust anchor information
	Entity B is authenticated using CSE-ID Certificate	Entity B's trust anchor information
	Entity B is authenticated using AE-ID Certificate	Entity B's trust anchor information
	Entity B is authenticated using Node-ID Certificate	Entity B's trust anchor information
MAF-Based		None

Mechanisms for Association Configuration of Entity A shall authenticate the configuration source and provide integrity protection for the configured information communicated from the configuration source to the entity.

9.1.2.1.2 Association Configuration of Entity B

Table 9.1.2.1.2-1 lists the parameters configured to the Registrar (Entity B) in the Association Configuration phase.

Table 9.1.2.1.2-1: Parameters configured to Entity B during the Association Configuration phase

Security Association Establishment Framework		Parameters specific to the Security Association Establishment Frameworks
Provisioned Symmetric Key		None
Certificate Based	Entity A is authenticated using Raw Public Key Certificate	None
	Entity A is authenticated using Device Certificate, CSE-ID Certificate, Node-ID Certificate or AE-ID Certificate	Entity A's trust anchor information
MAF-Based		None

Mechanisms for Association Configuration of Entity B shall authenticate the configuration source and provide integrity protection for the configured information communicated from the configuration source to the entity.

9.1.2.2 Association Configuration of M2M Authentication Functions

Table 9.1.2.2-1 lists the parameters configured to M2M Authentication Functions in the Association Configuration phase.

Table 9.1.2.2-1: Parameters configured to a M2M Authentication Functions during the Association Configuration phase

Security Association Establishment Framework		Parameter
MAF-Based	A-to-MAF Authentication	Entity B's CSE-ID or AE-ID (IdB)

The present document assumes that Association Configuration of the M2M Authentication Functions will utilize business logic of the Stakeholder that operates the M2M Authentication Function, and the details are not described in the present document.

9.2 Remote Security Provisioning Framework Procedures and Parameters

9.2.1 Bootstrap Credential Configuration Procedures and Parameters

9.2.1.0 Introduction

The following Bootstrap Credential Configuration procedures are described in this clause:

- Bootstrap Credential Configuration of Enrolees and Enrolment Targets (except for the GBA-Based case as discussed below), see clause 9.2.1.1.
- Bootstrap Credential Configuration of M2M Enrolment Functions (except for the GBA-Based case as discussed above), see clause 9.2.1.2.

The following Bootstrap Credential Configuration procedures are specified by other organizations:

- Bootstrap Credential Configuration of Underlying Network Service Provider authentication servers (e.g. HLR, HSS or AAA) for the GBA-Based Security Association Establishment Framework. These details are specified by ETSI TS 133 220 [13] and TIA TIA-1098 [14].
- Bootstrap Credential Configuration of Enrolees for the GBA-Based Security Association Establishment Framework. These details are specified by ETSI TS 133 220 [13] and TIA TIA-1098 [14].

9.2.1.1 Bootstrap Credential Configuration of Enrolee

Table 9.2.1.1-1 lists the parameters configured to Enrolees in the Bootstrap Credential Configuration phase for authentication with the M2M Enrolment Function in the Pre-Provisioned Symmetric Enrolee Key Remote Security Provisioning Framework and Certificate-Based Remote Security Provisioning Framework.

Table 9.2.1.1-1: Parameters configured to Enrolees during the Bootstrap Credential Configuration phase

Remote Security Provisioning Framework		Parameter
Pre-Provisioned M2M Secure Connection Key authentication. Not applicable to MAF.		Kpm
		KpmID
		MEF URI
Certificate-Based authentication	Enrolee authenticates itself using a raw public key	Enrolee's Private Key
		Enrolee's Raw Public Key Certificate
	Enrolee authenticates itself using a device certificate	Enrolee's Private Key
		Enrolee's Certificate and Chain
	Enrolee authenticates itself using a CSE-ID, Node-ID or AE-ID certificate	Enrolee's Private Key
		Enrolee's Certificate and Chain

The Bootstrap Credential Configuration of an Enrolee for the Pre-Provisioned Symmetric Enrolee Key Remote Security Provisioning Framework and Certificate-Based Remote Security Provisioning Framework shall authenticate the configuration source and shall provide confidentiality and integrity protection of the configured information communicated from the configuration source to the secured environment of the Enrolee. The present document does not specify any such mechanisms.

The Bootstrap Credential Configuration of an Infrastructure Domain Enrolment Target (including an M2M Authentication Functions) expected to use business logic of the Stakeholder operating the Infrastructure Domain Enrolment, and the details are not described in the present document.

9.2.1.2 Bootstrap Credential Configuration of M2M Enrolment Functions

It is assumed that an M2M Enrolment Function already knows its FQDN.

Table 9.2.1.2-1 lists the parameters configured to M2M Enrolment Functions in the Bootstrap Credential Configuration phase for mutual authentication with Enrolees and Enrolment Targets using the Pre-Provisioned Symmetric Enrolee Key Remote Security Provisioning Framework and Certificate-Based Remote Security Provisioning Framework.

Table 9.2.1.2-1: Parameters configured to the M2M Enrolment Function during the Bootstrap Credential Configuration phase for mutual authentication with Enrolees and Enrolment Targets using the Pre-Provisioned Symmetric Enrolee Key Remote Security Provisioning Framework and Certificate-Based Remote Security Provisioning Framework

Remote Security Provisioning Framework	Parameters specific to the Remote Security Provisioning Frameworks
Pre-Provisioned Symmetric Enrolment Key authentication of Enrolee or Enrolment Target	Kpm
	KpmID
Certificate Based authentication of Enrolee or Enrolment Target	MEF Private Key
	MEF Certificate and Chain

The Bootstrap Credential Configuration of M2M Enrolment Functions is expected to use business logic of the stakeholder operating the M2M Enrolment Function, and the details are not described in the present document.

9.2.2 Bootstrap Instruction Configuration Procedures and Parameters

9.2.2.0 Introduction

The following Bootstrap Instruction Configuration procedures are described in this clause:

- Bootstrap Instruction Configuration of Enrolees, see clause 9.2.2.1.
- Bootstrap Instruction Configuration of M2M Enrolment Functions, see clause 9.2.2.3.
- Bootstrap Instruction Configuration of Underlying Network Service Provider authentication servers (e.g. HLR, HSS or AAA), see clause 9.2.2.4.

9.2.2.1 Bootstrap Instruction Configuration of Enrolees

Table 9.2.2.1-1 lists the parameters configured to an Enrolee during the Bootstrap Instruction Configuration phase which are common to all Remote Security Provisioning Frameworks.

Table 9.2.2.1-1: Parameters configured to an Enrolee during the Bootstrap Instruction Configuration phase of which are common to all Remote Security Provisioning Frameworks

Parameter common to all Remote Security Provisioning Frameworks
Enrolment Target Identifier (Enrolee B's AE-ID or CSE-ID, or MAF-ID)

Table 9.2.2.1-2 lists the Remote Security Provisioning Framework-specific parameters configured an Enrolee in the Bootstrap Instruction Configuration phase of the Remote Security Provisioning Framework.

Table 9.2.2.1-2: Remote Security Provisioning Framework - specific parameters configured to an Enrolee during the Instruction Configuration phase of the Remote Security Provisioning Framework

Remote Security Provisioning Framework	Remote Security Provisioning Framework-specific Parameters
Pre-Provisioned Symmetric Enrolment Key	Enrolment Expiry
Certificate Based	MEF URI
	MEF Trust Anchor Information
GBA-Based	*None*

Mechanisms for Bootstrap Instruction Configuration of Enrolees shall authenticate the configuration source and shall provide at least integrity protection of the configured information communicated from the configuration source to the Enrolee.

9.2.2.2 Void

9.2.2.3 Bootstrap Instruction Configuration of M2M Enrolment Functions

Table 9.2.2.3-1 lists the parameters configured to an M2M Enrolment Function during the Bootstrap Instruction Configuration phase which are common to the Pre-Provisioned Symmetric Enrolee Key Remote Security Provisioning Framework and Certificate-Based Remote Security Provisioning Framework.

Table 9.2.2.3-1: Parameters configured to M2M Enrolment Functions during the Bootstrap Instruction Configuration phase which are common to the Pre-Provisioned Symmetric Enrolee Key Remote Security Provisioning Framework and Certificate-Based Remote Security Provisioning Framework

Parameter common to all Remote Security Provisioning Frameworks
Enrolment Target Identity (Enrolee B's CSE-ID or AE-ID, or MAF-ID)

Table 9.2.2.3-2 lists the Remote Security Provisioning Framework-specific parameters configured to an M2M Enrolment Functions in the Bootstrap Instruction Configuration phase of the Pre-Provisioned Symmetric Enrolee Key Remote Security Provisioning Framework and Certificate-Based Remote Security Provisioning Framework.

Table 9.2.2.3-2: Remote Security Provisioning Framework-specific parameters configured to an M2M Enrolment Function during the Instruction Configuration phase of the Pre-Provisioned Symmetric Enrolee Key Remote Security Provisioning Framework and Certificate-Based Remote Security Provisioning Framework

Remote Security Provisioning Framework		Remote Security Provisioning Framework-specific Parameters
Pre-Provisioned Symmetric Enrolment Key		Enrolment Expiry
Certificate Based	Enrolee is authenticated using a raw public key certificate	Enrolee's Public key identifier
	Enrolee is authenticated using a device certificate	Enrolee's M2M Device ID
		Enrolee's Trust Anchor Information
	Enrolee is authenticated using a CSE-ID, Node-ID or AE-ID certificate	Enrolee's Trust Anchor Information

The present document assumes that Bootstrap Instruction Configuration of the M2M Enrolment Functions utilizes business logic of the Stakeholder that operates the M2M Enrolment Function, and the details are not described in the present document.

9.2.2.4 Bootstrap Instruction Configuration of UNSP Authentication Server

Table 9.2.2.4-1 lists the parameters configured to an Underlying Network Service Provider authentication server (e.g. HLR, HSS or AAA) during the Bootstrap Instruction Configuration phase of the GBA-Based Remote Security Provisioning Framework.

Table 9.2.2.4-1: Parameters configured to M2M Enrolment Functions during the Bootstrap Instruction Configuration phase of the GBA-Based Remote Security Provisioning Framework

Parameter	Mandatory/Optional for all Remote Security Provisioning Frameworks
Enrolment Target Identifier (Enrolee B's CSE-ID or AE-ID, or MAF-ID)	Mandatory

The Bootstrap Instruction Configuration of the Underlying Network Service Provider authentication server is achieved by updating the GBA User Security Settings (GUSS) (ETSI TS 133 220 [13]) of the User Equipment (UE) upon which the Enrolee is executed. The present document assumes that this Bootstrap Instruction Configuration utilizes business logic of the Underlying Network Service Provider, and the details are not described in the present document.

9.2.3 End-to-End Credential Configuration Procedures and Parameters

9.2.3.0 Introduction

The following End-to-End Credential Configuration procedures are described in this clause:

- End-to-End Credential Configuration of Source ESF End-Points and Target ESF End-Points, see clause 9.2.3.1.

- End-to-End Credential Configuration of Trust Enabling Functions, see clause 9.2.3.2.
- Configuration parameters for enabling End-to-End Security at Source ESF End-Points and Target ESF End-Points, see clause 9.2.3.3.

9.2.3.1 End-to-End Credential Configuration of Source ESF End-Points and Target ESF End-Points

It is assumed that the Source ESF End-Point and the Target ESF End-Points are configured with the URI of the Trust Enabling Function and have been configured with the appropriate parameters specific to the Remote Security Provisioning Frameworks as described in clause 9.2. In addition, the end-to-end credentials are provisioned and appropriate security parameters are provisioned to the Target ESF End-Points while the Source ESF End-Point can derive the end-to-end credentials on its own using the relevant security parameters that have been provisioned. Table 9.2.3.1-1 provides a list of the parameters.

Table 9.2.3.1-1: Security Credentials and parameters provisioned to the Target ESF End-Points and Source ESF End-Points

Security Protection	End-to-End Security Provisioning Framework Parameters	Description
End-to-End Security Credentials	KpsaID	This is the provisioned credential-Id of the M2M Provisioned Symmetric Key.
	Kpsa	This is the M2M Provisioned Symmetric Key. This is used to derive the end-to-end master secret, Ke2e_master as described in clause 10.3.6.
	TEF URI	The URI of the trusted-third-party (TEF) entity that is used as the credential generator/registry and enables the registration and generation of end-to-end security credentials.
Cryptographic Parameters	Salt	The salt used for generating the end-to-end credentials. Optional parameter.
	Key Extraction Algorithm: HMAC-Hash	The Key extraction algorithm that is used for generating the various keys shall follow the mechanisms described in [48].
	Cryptographic Labels	The labels that are used by the cryptographic algorithms. The labels shall be used according to clause 10.3.6.1.
Types of Credentials	Message Authenticity (Primitive)	The key used for message authentication and integrity of oneM2M primitives. If the keying material is provided then it is generated by the ESF Target End-Point.
	Message Confidentiality (Primitive)	The key that is used for message confidentiality of oneM2M primitives. If the keying material is provided then it is generated by the Target ESF End-Point.
	Integrity of Data (Attribute)	Key used for providing integrity of data/attribute. If the keying material is provided then it is generated by the Target ESF End-Point.
	Confidentiality of Data (Attribute)	Key used for providing confidentiality of data/attributes. If the keying material is provided then it is generated by the Target ESF End-Point.

9.2.3.2 End-to-End Credential Configuration at the M2M Trust Enabling Functions

It is assumed that the Trust Enabling Function is configured with the identities of the entities (ESF Source and Target End-Points) and appropriate parameters specific to the Remote Security Provisioning Frameworks as described in clause 9.2.

In addition, the Trust Enabling Function is provisioned with the appropriate security parameters, so that the End-to-End security credentials can be derived and the set of the cryptographic parameters can be provisioned to the Target ESF End-Points once the Target ESF End-Point has been authenticated. Table 9.2.3.2-1 provides a list of the parameters.

Table 9.2.3.2-1: Security Parameters provisioned at the M2M Enrolment or Trust Enabling Function and Source ESF End-Point

End-to-End Security Protection	End-to-End Security Provisioning Framework Parameters	Description
End-to-End Security Credentials	Kpm	Pre-provisioned credentials between the Source ESF End-Point and TEF
	KpmID	The credential identity of the pre-provisioned credentials
	Source ESF End-Point identity (AE-ID/CSE-ID) Target ESF End-Point identity (CSE-ID)	The entity identity that is pre-provisioned with the end-to-end security credentials
List of required Security Protection and Strength	Message Authentication: (Low - High)	Provides a level of the required strength of the message authentication mechanism
	Message Confidentiality: (Low - High)	Provides a level of the required strength for providing message confidentiality mechanism
	Attribute Integrity: (Low - High)	Provides a level of the required strength for providing attribute integrity
	Attribute Confidentiality: (Low - High)	Provides a level of the required strength for providing attribute confidentiality

9.2.3.3 Configuration parameters for enabling End-to-End Security at Source ESF End-Points and Target ESF End-Points

The Source ESF End-Points and the Target ESF End-Points are provisioned with the cryptographic parameters that are used to enable and verify end-to-end security protection. In the case of the Target ESF End-Point, the Trust Enabling Function provisions the parameters to it after a successful authentication and derivation of the Secure Connection Key (Kpsa). In the case of the Source ESF End-Point, the parameters may have been pre-configured or provisioned in a similar manner as the Target ESF End-Point, that is, once the derivation of the Secure Connection Key (Kpsa) is done, and shared between the Source ESF End-Point and the Target ESF End-Point. Table 9.2.3.3-1 provides a list of the parameters.

Table 9.2.3.3-1: Security Parameters provisioned to the Target ESF End-Point and the Source ESF End-Point

End-to-End Security Protection	End-to-End Security Provisioning Framework Parameters	Description
End-to-End Security Credentials	e2e_master	The End-to-End master credential
	E2EKeyId	End-to-End Master credential identity
	Target ESF End-Point Identity (CSE-ID) Source ESF End-Point Id (AE-ID/CSE-ID)	The identity of the end entity with which the end-to-end credential is associated with
Cryptographic Parameters	Protocol: JWS/JWE, XML Sec	The type of encoding and representation that is used
	Class of cryptographic algorithms: AEAD (single key) or non-AEAD	Defines the class of cryptographic algorithms that shall be used
	Message Authenticity Algorithm/Size: HMAC-SHA-256, HMAC-SHA-512	Indicates the message authentication algorithm and key size
	Message Confidentiality Algorithm/Size: AES-192/256	Indicates the message confidentiality algorithm and key size
	Attribute Confidentiality Algorithm: AES-192/256	Attribute confidentiality algorithm and key size
	Attribute Authenticity Algorithm/Size: HMAC-SHA-256	Attribute authenticity and integrity algorithm and key size
Cryptographic Usage	Message/Attribute Authenticity: Nonce	The random value that was used for providing freshness. This is only stored temporarily associated with an expiration time and communicated to the other end
	Message/Attribute Confidentiality: Initialization Vector	This random value that is used as the initialization vector for the confidentiality algorithm

NOTE: For AEAD class of algorithms where only a single key is used, then only a single key would be generated and an associated cryptographic algorithm (e.g. AES-GCM or AES-CCM) identified. In addition, for AEAD class of algorithms, both an IV and a Nonce would not be generated, rather only a single random value, Nonce, would be generated.

10 Protocol and Algorithm Details

10.1 Certificate-Based Security Framework Details

10.1.1 Certificate Profiles

10.1.1.0 General

NOTE: These certificate profiles are compliant with the CoAP specification IETF RFC 7252 [i.21].

10.1.1.1 Common Certificate Details

All certificates shall conform to the following profile:

- Certificates shall conform to IETF RFC 5280 [34].
- The certificate shall include a SubjectPublicKeyInfo that indicates an algorithm of id-ecPublicKey with namedCurves secp256r1 [34]; this curve is equivalent to the NIST P-256 curve [39].
- The public key format shall be uncompressed [46].
- The hash algorithm shall be SHA-256.
- The key usage extension shall be included and shall indicate at least digitalSignature.

10.1.1.2 Raw Public Key Certificate Profile

Raw public key certificates shall conform to clause 10.1.1.1 and IETF RFC 7250 [37].

10.1.1.3 Details Common to Certificates with Certificate Chains

Certificates with Certificate Chains shall conform to the following description:

- These certificates shall conform to clause 10.1.1.1.
- Certificates shall be signed with ECDSA using secp256r1 or optionally RSA using at least 2048 key length, and the signature shall use SHA-256.
- Certificate chains should limit the number of intermediate CA certificates to avoid having a negative impact in constrained environments.

10.1.1.4 Profile for Device Certificates and their Certificate Chains

10.1.1.4.1 Profile for Device Certificates

Device certificates shall conform to the following description:

- Device certificates shall conform to clause 10.1.1.3.
- The subjectAltName extension of device certificates shall include one or more globally unique hardware instance identifiers.

EXAMPLE: Annex H "Object Identifier Based M2M Device Identifier" ETSI TS 118 101 [1] defines an object identifier -based M2M Device ID that can be used for providing a one or more globally unique hardware instance identifier. An object identifier -based M2M Device ID can be representing in an otherName field in the subjectAltName extension, where:

- otherName "type-ID" component is set to the M2M Device Indication ID (clause H.2.1 ETSI TS 118 101 [1]) arc of the object identifier M2M Device ID; and
- the otherName "value" component is set to the remainder of the object identifier M2M Device ID: Manufacturer ID arc, Model ID arc, Serial Number ID arc and optional Expanded ID arc (see clause H.2 ETSI TS 118 101 [1]).

NOTE: Providing the Model ID as part of the M2M Device ID can have privacy implications in some scenarios.

10.1.1.4.2 Profile for Certificate Authority Certificates for Device Certificates

Certificate Authority Certificates in the certificate chain for a device certificate shall conform to the following description:

- These certificates shall conform to clause 10.1.1.3.
- Certificate Authority Certificates for device certificates are recommended to use the name constraints extension (see clause 4.2.1.10 of IETF RFC 5280 [34]) to constrain the globally unique hardware instance identifiers in subsequent device certificates in a certification path.

EXAMPLE: Name constraints are defined in terms of permitted or excluded name subtrees. Subtrees of an object identifier based M2M Device ID name space are represented by an otherName field with:

- "type-ID" set to the M2M Device Indication ID (clause H.2.1 ETSI TS 118 101 [1]) arc of the applicable object identifier M2M Device ID name space; and
- "value" set to set to the remainder of the object identifier identifying the subtree.

10.1.1.5 Profile for AE-ID Certificates and their Certificate Chains

AE-ID certificates and all other certificates in the corresponding certificate chain shall conform to clause 10.1.1.3.

The full URI representation of the AE-ID shall be included in the subjectAltName extension.

The certificate used to sign the AE-ID certificate shall include nameConstraints satisfied by the hostname part of the full URI representation of the AE-ID.

AE-ID certificates shall not include wildcards.

10.1.1.6 Profile for FQDN Certificates and their Certificate Chains

FQDN Certificates and all other certificates in the corresponding certificate chain shall conform to clause 10.1.1.3.

An FQDN Certificate shall include the FQDN of the subject M2M Enrolment Function in the subjectAltName extension.

FQDN Certificates shall not include wildcards.

10.1.1.7 Profile for CSE-ID Certificates and their Certificate Chains

CSE-ID certificates and all other certificates in the corresponding certificate chain shall conform to clause 10.1.1.3.

The subjectAltName extension shall include the public domain name representation of the CSE-ID as defined in ETSI TS 118 101 [1].

CSE-ID certificates shall not include wildcards.

10.1.1.8 Profile for Node-ID Certificates and their Certificate Chains

Node-ID certificates and all other certificates in the corresponding certificate chain shall conform to clause 10.1.1.3.

The subjectAltName extension shall include the Node-ID as defined in ETSI TS 118 101 [1].

Node-ID certificates shall not include wildcards.

10.1.2 Public Key Identifiers

The public key identifier for a raw public key certificate shall calculated as described in section 2 of IETF RFC 6920 [40] using the SHA-256 hash algorithm. The public key identifier shall be generated using one of the sha-256-120, sha-256-128 or sha-256 hash algorithms specified in IETF RFC 6920 [40].

It is recommended that the public key identifier be as long as practical within the deployment constraints.

The trusted public key identifier (received during Association Configuration or Bootstrap Instruction Configuration) is matched against the raw public key certificate (received during the Security Handshake) using the following procedure:

1) A check digest value is computed according to section 2 of IETF RFC 6920 [40] using the hash algorithm identified in the trusted public key identifier.

2) The check digest value is compared against the digest value encoded in the trusted public key identifier. If the values are identical then the raw public key certificate matches the trusted public key identifier. Otherwise, the raw public key certificate does not match the trusted public key identifier.

10.1.3 Support Requirements for each Public Key Certificate Flavour

Table 10.1.3-1 lists, for each of the various types of entity (Field Domain CSE, Field Domain AE, IN-CSE, IN-AE, M2M Authentication Function and M2M Enrolment Function), the flavour of certificate that may be issued to the entity and the flavour of other entity's certificates that the entity is required to be able to process. In this table "O" indicates optional, "M" indicates Mandatory, "CA" indicates that the option is required if the entity supports the certificate-based security association establishment framework, "CB" indicates conditional on the entity supporting certificate-based Remote Security Provisioning framework.

Table 10.1.3-1: Applicability of certificate flavours issued to an entity and flavours of other entity's certificates that the entity is required to be able to process

Entity	Flavour of certificate may be issued to entity						Flavour of other entity's certificates that the entity is recommended to be able to process.					
	Raw	Device	CSE-ID	AE-ID	Node-ID	FQDN	Raw	Device	CSE-ID	AE-ID	Node-ID	FQDN
Field Domain CSE	O	O	O	-	O	-	CA	CA	CA	CA	CA	CB
Field Domain AE	O	O	-	O	O	-	CA	CA	CA	-	CA	CB
IN-CSE	O	-	O	-	-	-	CA	CA	CA	CA	CA	-
IN-AE	O	-	-	O	-	-	CA	-	CA	-	-	-
MAF	-	-	-	-	-	M	-	-	-	-	-	M
MEF	-	-	-	-	-	M	CB	CB	-	-	CB	M

Mutual authentication between remote management servers and remote management clients is not considered in the present document. Where supported, Remote Security Administration may be used to provision the certificates.

10.1.4 Certificate Signing Request Profile

A Certificate Signing Request (CSR) is a signed object provided to the Certificate Provisioning server (EST Server or SCEP Server) to request the issuing of a certificate. Certificate Provisioning as specified in clause 8.3.6 may be used to issue a certificate to an Node, CSE or AE. The certificate signing request shall include:

- the subjectPublicKeyInfo: the public key and the algorithm with which key is used;
- extensions:
  - subjectAltName: This field shall contain the AE-ID, CSE-ID or Node-ID using the name type defined for each type of certificate in clauses 10.1.1.5, 10.1.1.7 and 10.1.1.8.

The certificate signing request may include additional fields and extensions provided by the Certificate Provisioning server, for example using the EST Certificate Signing Request (CSR) Attributes Request described in section 2.6 of IETF RFC 7030 [59].

10.2 TLS and DTLS Details

10.2.1 TLS and DTLS Versions

Where TCP payloads are to be secured, TLS v1.2 [5] shall be used.

Where UDP payloads are to be secured, DTLS v1.2 [6] shall be used, noting that the DTLS v1.2 ciphersuites are identical to the TLS v1.2 ciphersuites.

All implementations shall support the Server Name Indication (SNI) to indicate their authority in the SNI HostName field as defined in section 3 of IETF RFC 6066 [44]. This is needed so that when a host that acts as a virtual server for multiple Authorities receives a new TLS or DTLS connection, it knows which keys to use for the TLS or DTLS session.

(D)TLS Clients on any Node and (D)TLS Servers on MNs shall support at least one of the TLS ciphersuites indicated in clause 10.2.2 or clause 10.2.3.

NOTE: (D)TLS Servers on MN need to support the TLS ciphersuites for those (D)TLS clients they are expected to interact with.

(D)TLS Servers on INs shall support all of the TLS ciphersuites indicated in clauses 10.2.2 and 10.2.3.

10.2.2 TLS and DTLS Ciphersuites for TLS-PSK-Based Security Frameworks

The following Security Frameworks:

- Provisioned Symmetric Key Security Association Establishment Framework;
- MAF-Based Security Association Establishment Framework;
- Pre-Shared Key Remote Security Provisioning Framework;
- GBA-Based Remote Security Provisioning Framework;

shall use one of the key exchange algorithms defined in IETF RFC 4279 [15].

TLS implementations in entities supporting these security frameworks shall implement at least the following TLS ciphersuite:

- TLS_PSK_WITH_AES_128_CBC_SHA256 (IETF RFC 5487 [42]).

DTLS implementations supporting these security frameworks shall implement at least the following ciphersuites

- TLS_PSK_WITH_AES_128_CCM_8 (IETF RFC 6655 [31]).

The security considerations of section 7 of IETF RFC 4279 [15] apply. In particular, applications should carefully weigh whether or not they need Perfect Forward Secrecy (PFS) and select an appropriate ciphersuite (section 7.1 of IETF RFC 4279 [15]).

10.2.3 TLS and DTLS Ciphersuites for Certificate-Based Security Frameworks

The following Security Frameworks:

- Certificate-Based Security Association Establishment Framework;
- Certificate-Based Security Bootstrap Framework;

shall use the standard TLS handshake (IETF RFC 5246 [5]) with the ECDHE_ECDSA Key Exchange (IETF RFC 8422 [43]).

TLS implementations supporting these security frameworks shall implement at least the following ciphersuite:

- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256, IETF RFC 5289 [32].

TLS implementations may support RSA keys (2048 bits or more):

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256, IETF RFC 5289 [32].

DTLS implementations supporting these security frameworks shall implement at least the following TLS ciphersuite:

- TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8, IETF RFC 7251 [45].

DTLS implementations may support RSA keys (2048 bits or more):

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256, IETF RFC 7525 [80].

Implementations supporting these security frameworks shall support authenticating other entities using all available public key certificate flavours (see clause 8.1.2.1):

- Raw public key certificate: using the mechanism specified in IETF RFC 7250 [37], Implementation shall support receiving and processing raw public keys compliant with section 9.1.3.2 in IETF RFC 7252 [i.21].
- All other certificates: X.509 certificates including device hardware identifier. Implementation shall support receiving and processing raw public keys compliant with section 9.1.3.3 in IETF RFC 7252 [i.21].

10.3 Key Export and Key Derivation Details

10.3.1 TLS Key Export Details

TLS Key Export Details for Enrolment Key

Following successful TLS authentication between the Enrolee and M2M Enrolment Function, see clause 8.3.1.2, the Enrolment Key (Ke) and RelativeKeID are generated from the (D)TLS session secrets by the Enrolee and M2M Enrolment Function by applying TLS Key Export (IETF RFC 5705 [18]) using the label "EXPORTER-oneM2M-Bootstrap" and length 48. The Enrolment Key (Ke) is set to the value of the 32 least significant bytes, while RelativeKeID is set to the value of the 16 most significant bytes.

TLS Key Export Details for M2M Secure Connection Key

Following successful TLS authentication between the Entity A and the M2M Authentication Function (MAF), see clause 8.8.2.7, the M2M Secure Connection Key (Kc) and the M2M Secure Connection Key Identifier (KcID) are generated from the (D)TLS session secrets by the Entity A and the MAF by applying TLS Key Export (IETF RFC 5705 [18]) using the label "EXPORTER-oneM2M-Connection" and length 48. The M2M Secure Connection Key (Kc) is set to the value of the 32 least significant bytes, while M2M Secure Connection Key Identifier (KcID) is set to the value of the 16 most significant bytes.

TLS Key Export Details for pairwiseE2EKey

Following successful TLS authentication between the ESCertKE Initiating End-Point and ESCertKE Terminating End-Point, see clause 8.7.2.2, the pairwiseE2EKey and pairwiseE2EKeyID are generated from the (D)TLS session secrets by the Enrolee and M2M Enrolment Function by applying TLS Key Export (IETF RFC 5705 [18]) using the label "EXPORTER-oneM2M-ESCertKE" and length 48. The pairwiseE2EKey is set to the value of the 32 least significant bytes, while pairwiseE2EKeyID is set to the value of the 16 most significant bytes.

10.3.2 Derivation of Master Credential from Enrolment Key

This clause describes the details when generating a Master Credential (Km) from an Enrolment Key (Ke) in Security Bootstrap Frameworks.

The following information shall be used when generating Km from Ke:

- the value of the Enrolment Key (Ke);
- the M2M Authentication Function Identifier (MAF-ID) shall be encoded to an octet string according to UTF-8 encoding rules as specified in IETF RFC 3629 [19] and apply Normalization Form KC (NFKC) as specified in [20].

The value of Km shall be generated as:

Km:= HMAC-SHA-256(Ke, "oneM2M Enrolment Key to Master Credential derivation" || MAF-ID),

where HMAC-SHA-256 is defined in IETF RFC 2104 [33].

10.3.3 Derivation of Provisioned Secure Connection Key from Enrolment Key

This clause describes the details when generating a Provisioned Secure Connection Key (Kpsa) from an Enrolment Key (Ke) in Remote Provisioning Frameworks.

The following information shall be used when generating Kpsa from Ke:

- The value of the Enrolment Key (Ke).
- Enrolee B's CSE-ID or AE-ID (Enrolee-B-ID), which shall be encoded to an octet string according to UTF-8 encoding rules as specified in IETF RFC 3629 [19] and apply Normalization Form KC (NFKC) as specified in [20].

The value of Kpsa shall be generated as:

- Kpsa:= HMAC-SHA-256(Ke, "oneM2M Enrolment Key to Provisioned Secure Connection Key derivation" || Enrolee-B-ID);

where HMAC-SHA-256 is defined in IETF RFC 2104 [33].

10.3.4 Generating KeID

The KeID value shall be formed as:

- KeID = hexBinary(RelativeKeID)@MEF-FQDN:

where:

- hexBinary(RelativeKeID) denotes the hexadecimal representation of the binary value of RelativeKeID; and
- MEF-FQDN denotes the FQDN of the M2M Enrolment Function.

10.3.5 Generating Key Identifier for the MAF Security Framework

The Key Identifier value shall be formed as:

- Key Identifier = RelativeKeyID@MAF-FQDN;

where:

- RelativeKeyID = hexBinary(KcID) denotes the hexadecimal representation of the binary value of KcID; and
- MAF-FQDN denotes the FQDN of the M2M Authentication Function.

10.3.6 Derivation of End-to-End Master Key from Provisioned Secure Connection Key

10.3.6.1 Introduction

This clause describes the details when generating an End-to-End Master Key (Ke2e_master) based on a successful establishment of security association between a Source ESF End-Point and Target ESF End-Point using a Remote Security Provisioning Framework as described in clause 8.3. The mechanisms to generate the End-to-End Master Key then uses a key extraction process using the Provisioned Secure Connection Key, (Kpsa).

The following information shall be used when generating Ke2e from Kpsa:

- The value of the Provisioned Secure Connection Key (Kpsa).
- Source ESF End-Point B's CSE-ID or AE-ID (Source ESF End-Point-B-ID), which shall be encoded to an octet string according to UTF-8 encoding rules as specified in IETF RFC 3629 [19] and applying Normalization Form KC (NFKC) as specified in [20].

The value of Ke2e_master shall be generated as:

- Ke2e_master = HMAC-Hash (Salt, Kpsa).

NOTE: In the case of Source-generated credentials, a random value generated by the Source ESF End-Point is used instead of the Kpsa in order to generate the Ke2e_master.

10.3.6.2 Key Extraction and Expansion of End-to-End Master Key

The End-to-End Master Key (Ke2e_master) is used to generate the security protection-specific keys. The Key Extraction and Expansion parameters along with the scope are used to generate the various keys. The Key extraction and expansion is performed according to the specifications defined in IETF RFC 5869 [48]. A list of possible End-to-End keys are shown in table 10.3.6.2-1.

Table 10.3.6.2-1: End-to-End Security Keys

Security Protection	Symmetric Keys Generated
Message Authenticity (Primitive)	Ke2e_msg_auth
Message Confidentiality (Primitive)	Ke2e_msg_conf
Integrity of Data (Attribute)	Ke2e_att_auth
Confidentiality of Data (Attribute)	Ke2e_att_conf

The End-to-End security protection keys that are generated by performing a key expansion of the Ke2e_master using mechanisms specified in IETF RFC 5869 [48]. Using the generated end-to-end master key, the associated end-to-end message authentication and or end-to-end message confidentiality keys and attribute keys are generated in the following manner:

- T(0) = empty string (zero length)
- End-to-End Message Authenticity Key (Ke2e_msg_auth) = T(1) = HMAC-Hash (Ke2e_master, T(0) | "E2E Message Authentication Key"| 0x01)
- End-to-End Message Confidentiality Key (Ke2e_msg_conf) = T(2) = HMAC-Hash (Ke2e_master, T(1)| "E2E Message Confidentiality Key"|0x02)
- End-to-End Attribute Authenticity Key (Ke2e_att_auth) = T(3) = HMAC-Hash (Ke2e_master, T(2)| "E2E Attribute Authenticity Key"|0x03)
- End-to-End Attribute Confidentiality Key (Ke2e_att_conf) = T(4) = HMAC-Hash (Ke2e_master, T(3)| "E2E Attribute Confidentiality Key"|0x04)

NOTE 1: If AEAD algorithms are used, where only a single key is used, then either the Ke2e_msg_auth or the Ke2e_msg_conf key may be derived and used for both message authenticity as well as message confidentiality.

NOTE 2: The Target ESF End-Point can be provisioned with all the required keys or can be provisioned only with the Master End-to-End key (Ke2e_master) and the associated cryptographic parameters (e.g. labels, random values) which are then used by the Target ESF End-Point in order to generate the keys required for ESPrim and ESData.

10.3.7 Derivation of Usage-Constrained Symmetric Keys from Enrolment Key

This clause describes the details when generating a usage-constrained symmetric key from an Enrolment Key (Ke) in Remote Security Provisioning Frameworks.

The following information shall be used:

- The value of the Enrolment Key (Ke).
- The Security Usage Identifier (SUID) applicable for the usage of the symmetric key.
- Enrolee Target's Identifier (Enrolment-Target-ID), which is an FQDN which shall be encoded to an octet string according to UTF-8 encoding rules as specified in IETF RFC 3629 [19] and apply Normalization Form KC (NFKC) as specified in [20]:
  - If the Enrolment Target is a CSE or AE, then the FQDN representation of the Absolute CSE-ID or Absolute AE-ID shall be used.

The value of the usage-constrained symmetric key shall be generated as:

- HMAC-SHA-256(Ke, "oneM2M Enrolment Key to Usage-Constrained Symmetric Key derivation" || SUID || Enrolment-Target-ID);

where HMAC-SHA-256 is defined in IETF RFC 2104 [33].

10.3.8 sessionESPrimKey Derivation Algorithms

10.3.8.1 Introduction

The sessionESPrimKey is used in End-to-End Security of Primitives (ESPrim), and derived from pairwiseESPrimKey, receiverESPrimRandObject, originatorESPrimRandObject, see clause 8.4.2.

Clause 10.3.8 specifies the algorithms used for derivation of the sessionESPrimKey used in ESPrim. The available algorithms are listed in table 10.3.8.1-1.

Table 10.3.8.1-1: sessionESPrimKey derivation algorithms

Algorithm	Mandatory/Optional	Clause
HMAC-SHA256	M	10.3.8.2

10.3.8.2 HMAC-SHA256 sessionESPrimKey Derivation Algorithm

The sessionESPrimkey is derived as

sessionESPrimKey = HMAC-SHA256 (pairwiseESPrimKey,
receiverESPrimRandObject || originatorESPrimRandObject ||
"oneM2M HMAC-SHA256 sessionESPrimKey derivation algorithm"),

where HMAC-SHA-256 is defined in IETF RFC 2104 [33].

10.4 Credential-ID Details

The Credential-ID has two parts:

- A type-ID part. The type-ID part is a positive integer defined by datatype sec:credIDTypeID.
- A value part which contains a globally-unique identifier for the entity's credential. The value part may use the Roman alphabet, numerals, '.', '_', '-', and '@'.

The Credential-ID is formed by concatenating the type part, the character '-' and the value part.

NOTE: A Credential-ID is a globally unique identifier used to identify *serviceSubscribedAppRule* resources (ETSI TS 118 101 [1]) and identify credentials in security configuration information.

10.5 KpsaID

The KpsaID shall be of the form:

- KpsaID = Issuer_Relative_KpsaID@Issuer-FQDN;

where:

- Issuer_Relative_KpsaID is composed of the Roman alphabet, numerals, '.', '_' and '-' characters. The issuer of KpsaID shall ensure that no two Kpsa have identical Issuer_Relative_KpsaID.
- Issuer-FQDN is an FQDN representing the stakeholder that provisioned Kpsa.

NOTE: This format for KpsaID allows the identity of the Issuer to be extracted from KpsaID.

10.6 KmID Format

The KmID shall be of the form:

- KmID = MAF_RELATIVE_KmID@MAF-FQDN;

where:

- MAF_RELATIVE_KmID is composed of the Roman alphabet, numerals, '.', '_' and '-' characters. The MAF_RELATIVE_KmID is not case sensitive. The MAF shall ensure that no two Km have identical MAF_RELATIVE_KmID.
- MAF-FQDN denotes the FQDN of the M2M Authentication Function.

NOTE: This format for KmID allows the identity of the M2M Authentication Function to be extracted from KmID.

10.7 Enrolment Expiry

Enrolment Expiry is the life time to be applied for the key generated, i.e. Ke as part of the Pre-Provisioned Symmetric Key Remote Security Provisioning. Keys that are generated for establishing security associations between Enrolees and the Enrolment Targets (i.e. Km or Kpsa) based upon the enrolment key Ke will not be valid after the lifetime expiration of the enrolment credential Ke. Therefore at the maximum, the lifetime of Km or Kpsa should be set to the lifetime associated with Ke. Once the Enrolment Expiry is exceeded, the Enrolee has to re-initiate remote provisioning to re-generate keys as described in the Remote Security Provisioning Frameworks as described in clause 8.3.2.1.

11 Privacy Protection Architecture using Privacy Policy Manager（PPM)

11.1 Introduction

This clause provides an architecture for the Privacy Policy Manager (PPM). PPM is a distributed authorization privacy protection architecture using M2M Service Subscriber's privacy preference and service's privacy policy.

The PPM is a personal data management framework based on the M2M Service Subscriber's privacy preferences and creates access control information from policies agreed by a M2M Service Subscriber. The PPM protects M2M Service Subscriber's personal data from unauthorized parties and unauthorized collection. The PPM may be operated by the M2M Service Provider itself or another entity on behalf of the M2M Service Provider.

If the M2M Service Provider provides M2M Service Subscriber's personal data to any third party, the M2M Service Provider needs to get the M2M Service Subscriber's consent. In case that the M2M Service Subscriber accepted a privacy policy which indicates provision to third party, the Service Provider could provide the personal data to third party. However, if the privacy policy did not include provision to third party, the Service Provider needs to update the privacy policy and get the M2M Service Subscriber's consent to it.

11.2 Components of PPM

11.2.1 Privacy Preference and Privacy Policy

The PPM shall manage privacy preferences and privacy policies.

- Privacy preference:
  - Privacy preference is M2M Service Subscriber's preference regarding the provision of his own personal data to third parties.
  - M2M Service Subscriber creates a M2M Service Subscriber's privacy preference and registers it to the PPM.
  - List of privacy attributes is described in annex J.

- Privacy policy:
  - Privacy policy describes a required personal data to provide a service to an M2M Service Subscriber by a M2M Service Provider.
  - A M2M Service Provider creates a privacy policy and registers it to the PPM.

11.2.2 Functions of PPM

The PPM may comprise the following functions.

- Sophisticated consent mechanism for matching a M2M Service Subscriber's privacy preference with the ASP's privacy policy:
  - Privacy policy describes required personal data to provide a service to a M2M Service Subscriber by a M2M Service Provider.
  - When the M2M Service Subscriber subscribes to a service which is provided by a M2M Service Provider, the M2M Service Provider needs to get the M2M Service Subscriber's consent to the service's privacy policy. The PPM provides friendly consent mechanism for M2M Service Subscriber by comparing the privacy preference and the privacy policy.
  - This function is described in clause 11.4.1.2.
- Functions of the Policy Decision Point (PDP), Policy Retrieval Point (PRP) for Distributed Authorization, management of AccessControlPolicy resources and Dynamic Authorization System (DAS) Server.
  - PDP:
    - When an Originator requests personal data from a Hosting CSE which acts as PEP, the Hosting CSE requests access control decision from the PPM which acts as PDP. The PPM creates <*authorizationDecision*> from access control information that is created from policies agreed by M2M Service Subscriber and respond <*authorizationDecision*> to the Hosting CSE.
    - Details of PDP and <*authorizationDecision*> are described in the present document, clause 7.5.2 and ETSI TS 118 101 [1], clause 9.6.42, respectively.
  - PRP:
    - When an Originator requests personal data from a Hosting CSE which acts as PDP, the Hosting CSE requests access control policies from the PPM which acts as PRP. The PPM creates <*authorizationPolicy*> based on policies agreed by M2M Service Subscribers and respond <*authorizationPolicy*> to the Hosting CSE.
    - Details of PRP and <*authorizationPolicy*> are described in the present document, clause 7.5.3 and ETSI TS 118 101 [1], clause 9.6.43, respectively.
  - Management of <*accessControlPolicy*> resources hosted by CSEs:
    - When an Originator requests personal data from a Hosting CSE, and this CSE uses locally stored <*accessControlPolicy*> resource to derive the access control decision using the mechanism described in clause 7.1, the PPM may act as an IN-AE which generates and deploys the required <*accessControlPolicy*> resources on the respective CSE and assigns appropriate *accessControlPolicyID* attributes to resources created by the M2M Service Subscriber.
  - Dynamic Authorization System:
    - Direct Dynamic Authorization
      - When an Originator requests personal data from an Hosting CSE, the Hosting CSE requests *dynamicACPInfo* or <*token*> from the PPM. The PPM creates *dynamicACPInfo* or <*token*> based on policies agreed by M2M Service Subscribers and respond *dynamicACPInfo* or <*token*> to the Hosting CSE.
      - Detail of Direct Dynamic Authorization is described in the present document, clause 7.3.2.2.

        - Details of *dynamicACPInfo* and <*token*> are described in ETSI TS 118 101 [1], clauses 9.6.40 and clause 9.6.39, respectively.

    - Indirect Dynamic Authorization:

        - Before an Originator requests personal data from a Hosting CSE, the Originator requests <*token*> or *tokenID* from the PPM. The PPM creates <*token*> based on policies agreed by M2M Service Subscribers and respond <*token*> or *tokenID* to the Originator. Then, the Originator requests personal data from Hosting CSE with <*token*> or *tokenID*.

        - Detail of Indirect Dynamic Authorization is described in the present document, clause 7.3.2.3

- Traceability of personal data usage:

    - PPM shall store the access log that records which Originator accessed which kind of collected data.

    - This function is for further study of oneM2M, but this function can be implemented using components that are defined in oneM2M.

11.3 Privacy Policy Management Architecture

11.3.1 Introduction

The PPM manages M2M Service Subscriber's preference and service's privacy policy of an M2M Application Service Provider. Basically, one M2M Application Service Provider have one PPM in infrastructure domain and manage status of M2M Service Subscriber consent in the PPM. There are four procedures in the use of the PPM. This clause explains relationships between steps in the PPM scenario and components of oneM2M:

1) A M2M Service Subscriber subscribes to services provided by the M2M Service Provider.

2) The M2M Service Subscriber subscribes to a service offered by an ASP. This may happen concurrently with step 1.

3) An AE (IN-AE or field domain AE) requests personal data that are stored in a Hosting CSE.

4) The M2M Service Subscriber checks the access log of his/her own personal data and requests the deletion of the collected personal data from the Hosting CSE.

11.3.2 Involved Entities

- M2M Service Subscriber:

    - An M2M Service Subscriber can make use of M2M services by subscribing to a service of an ASP which provides functions that control access to information handled by the M2M Service Provider.

    - When an M2M Service Subscriber subscribes to service provided by an ASP, the M2M Service Subscriber becomes a data subject.

- Personal Data:

    - Personal data is information that can be used on its own, or with other information to identify an individual to form Personally Identifiable Information (PII).

    - A Hosting CSE collects and stores personal data.

    - Examples of personal data: Sensor data, Electrical power consumption, Operating state of air conditioner, etc.

- ADN-AE, ASN-AE:

    - An ADN-AE or ASN-AE produces various kinds of data, such as sensor data. An ADN-AE or ASN-AE may also request data from resource hosting CSEs.

- The ADN-AE or ASN-AE sends the data to a Hosting CSE such as ASN-CSE, MN-CSE or IN-CSE

- Hosting CSE:
  - If the Hosting CSE use the PPM as PDP, the Hosting CSE should act as Policy Enforcement Point (PEP).
  - If the Hosting CSE use the PPM as PRP, the Hosting CSE should act as PDP.
  - If the Hosting CSE use the PPM as DAS, the Hosting CSE should configure <*dynamicAuthorizationConsultation*> resources linked to the requested resource:
    - <*dynamicAuthorizationConsultation*> resource is described in ETSI TS 118 101 [1], clause 9.6.40.

- Application Service Provider:
  - An Application Service Provider provides services to an M2M Service Subscriber who joins the M2M Service Provider.
  - An Application Service Provide requests personal data from an M2M Service Provider in order to provide services.

- M2M Service Provider:
  - M2M Portal:
    - An M2M portal provides a M2M Service Subscriber Interface through which services provided by an M2M Platform may be managed.
    - A M2M Service Subscriber accesses the M2M portal to subscribe to a service offered by an ASP.

- PPM:
  - The PPM may include functionality for an automated procedure (not defined by oneM2M) to create access control policies and to deploy these on CSEs according to the policies and the preferences agreed by a M2M Service Subscriber.
  - If the PPM acts as PDP or PRP, it requires CSE functionality. If the PPM acts as DAS Server, the PPM requires AE functionality.
  - The PPM may provide a M2M Service Subscriber Interface via a PPM portal. A M2M Service Subscriber may access the PPM portal to configure the M2M Service Subscriber's privacy preference. The PPM portal is out of scope of oneM2M.

11.3.3 Management Flow in PPM Architecture

11.3.3.0 Introduction

This clause describes the case where a M2M Service Provider stores personal data.

11.3.3.1 Subscribe to a M2M Service Provider

When a M2M Service Subscriber subscribes to a M2M Service Provider, the M2M Service Subscriber configures privacy preferences using the PPM. A privacy preference / policy explains what data is intended to be used by ASPs and allowed by consent to be shared with other service subscribers. Figure 11.3.3.1-1 illustrates this process.

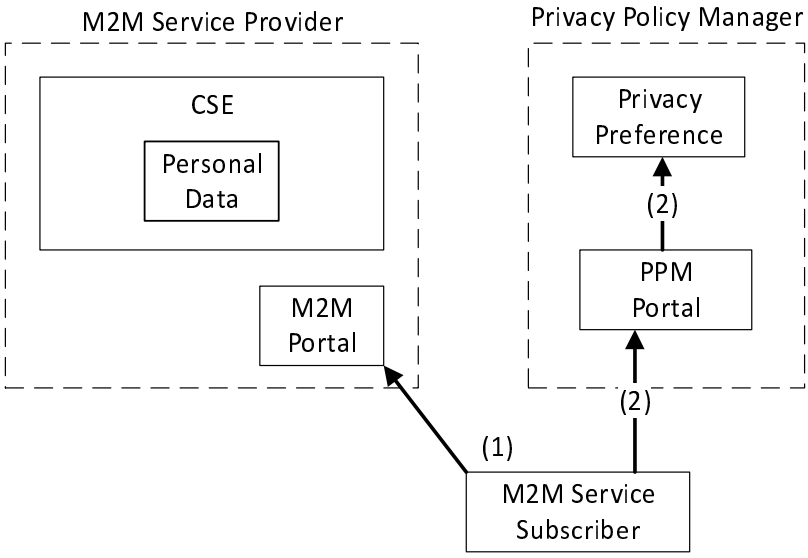


Figure 11.3.3.1-1: A M2M Service Subscriber subscribes to a M2M Service Provider

1. A M2M Service Subscriber accesses the M2M portal of a M2M Service Provider:
   - This process typically uses Web access protocols such as HTTP, HTTPS and so on.
   - This process is described in clause 11.4.1.2.
2. The M2M Service Subscriber configures a privacy preference and registers it on the PPM portal:
   - The M2M Service Subscriber accesses the PPM portal, or the M2M portal redirects the M2M Service Provider to the PPM portal. This process uses Web access protocols.
   - This process is described in clause 11.4.1.2.
3. The M2M Service Provider collects and stores data from AEs.

11.3.3.2 Subscription to a service by ASP

The M2M Service Subscriber can subscribe to various kinds of services provided by ASPs through the M2M Service Provider. Service lists are registered on an M2M portal and the M2M Service Subscriber can select services to subscribe to. When the M2M Service Subscriber subscribes to a service, the M2M Service Subscriber needs to accept ASP's privacy policy. In order for the M2M Service Subscriber to easily understand this policy, the PPM shall create the customized privacy policy based on the privacy policy provided by the ASP and the M2M Service Subscriber's privacy preference. Therefore, the M2M Service Subscriber can control personal data and agreement implies understanding of the privacy policy. Figure 11.3.3.2-1 shows the overview of this process.

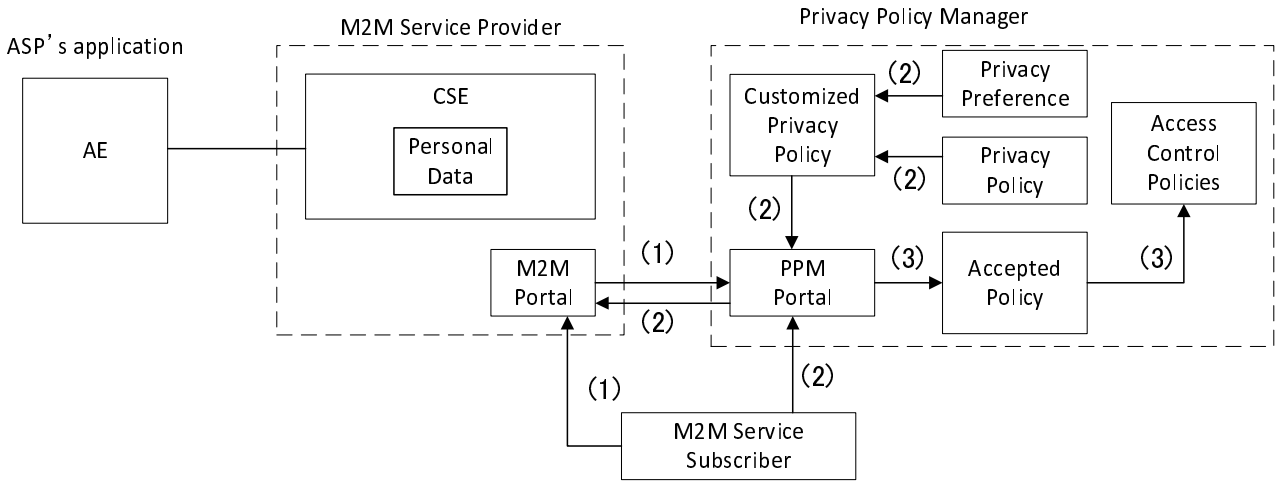


Figure: 11.3.3.2-1: The M2M Service Subscriber subscribes to an ASP's service

1. The M2M Service Subscriber accesses the M2M portal and selects an ASP's service to subscribe. The M2M portal redirects to the PPM portal to get the M2M Service Subscriber's consent:

   - This process typically uses Web access protocols such as HTTP, HTTPS and so on.

   - This process is described in clause 11.4.1.1.

2. The M2M Service Subscriber needs to accept a privacy policy to subscribe to the ASP's service. The PPM shall create the customized privacy policy for each M2M Service Subscriber based on the M2M Service Subscriber's privacy preference and service's privacy policy. It is easy for the M2M Service Subscriber to confirm differences between the privacy preference and the privacy policy and to understand what kind of personal data are collected by the ASP. After the M2M Service Subscriber accepts the privacy policy, the M2M Service Subscriber can subscribe to the ASP's service:

   - The function of creating a customized privacy policy is described in clause 11.4.1.3.

3. The PPM shall create or update access control policies using the privacy policy that the M2M Service Subscriber accepted:

   - The function of creating or updating access control policies in the PPM may rely on the authorization mechanisms specified in clauses 7.3 and 7.5. The details of the synchronization process are not specified in the present document.

11.3.3.3 Request for personal data to the Hosting CSE

11.3.3.3.1 Implementation options

When the ASP collects personal data to provide the service, it requests the personal data from a Hosting CSE in the M2M Service Provider's domain. Access to the personal data is controlled by the PPM which may work either as PDP, PRP or DAS Server as detailed below:

- If the PPM works as PDP, the Hosting CSE acts as PEP and requests <*authorizationDecision*> from the PPM and controls the data access using them. Figure 11.3.3.3.2-1 illustrates this process.

- If the PPM works as PRP, the Hosting CSE acts as PDP and requests <*authorizationPolicy*> from the PPM and controls the data access using them. Figure 11.3.3.3.3-1 illustrates this process.

- If the PPM works as DAS Server (Direct dynamic authorization), the Hosting CSE checks <*dynamicAuthorizationConsultation*> and requests *dynamicACPInfo* or <*token*> from the PPM. Figure 11.3.3.3.4.1-1 illustrates this process.

- If the PPM works as DAS Server (Indirect dynamic authorization), the ASP requests <*token*> or tokenID from the PPM. Figure 11.3.3.3.4.2-1 illustrates this process. (Detail of this request is not specified in oneM2M).

11.3.3.3.2 Option 1: PPM works as PDP

For this option, the PPM shall be implemented as CSE and shall provide an interface that enables access control for personal data using the PPM as PDP.

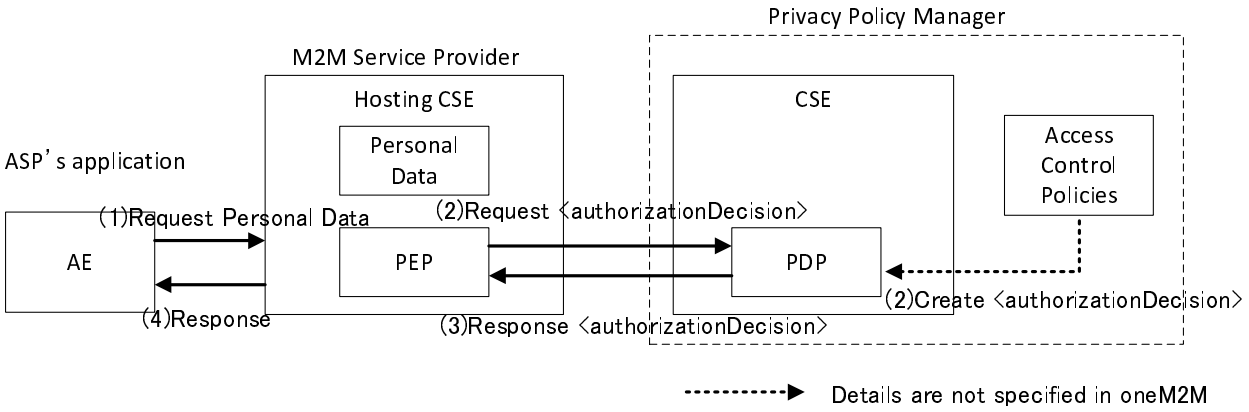


Figure 11.3.3.3.2-1: Request for personal data to the Hosting CSE (the PPM works as PDP)

1. The ASP requests personal data from the Hosting CSE in M2M Service Provider.
2. PEP in the Hosting CSE requests <*authorizationDecision*> from the PPM. The PPM shall create <*authorizationDecision*> using access control policies.

   The PPM could use <*accessControlPolicy*> resources as access control policies. In this case, CSE in the PPM stores <*accessControlPolicy*>.

   Detail of creating <*authorizationDecision*> from access control policies is not specified in oneM2M.
3. The PPM shall respond <*authorizationDecision*> to the Hosting CSE.
4. If accessing personal data is permitted, the Hosting CSE accesses the personal data and sends the personal data to the ASP as a response.

11.3.3.3.3 Option 2: PPM works as PRP

For this option, the PPM shall be implemented as CSE and shall provide an interface that enables access control for personal data using the PPM as PRP.

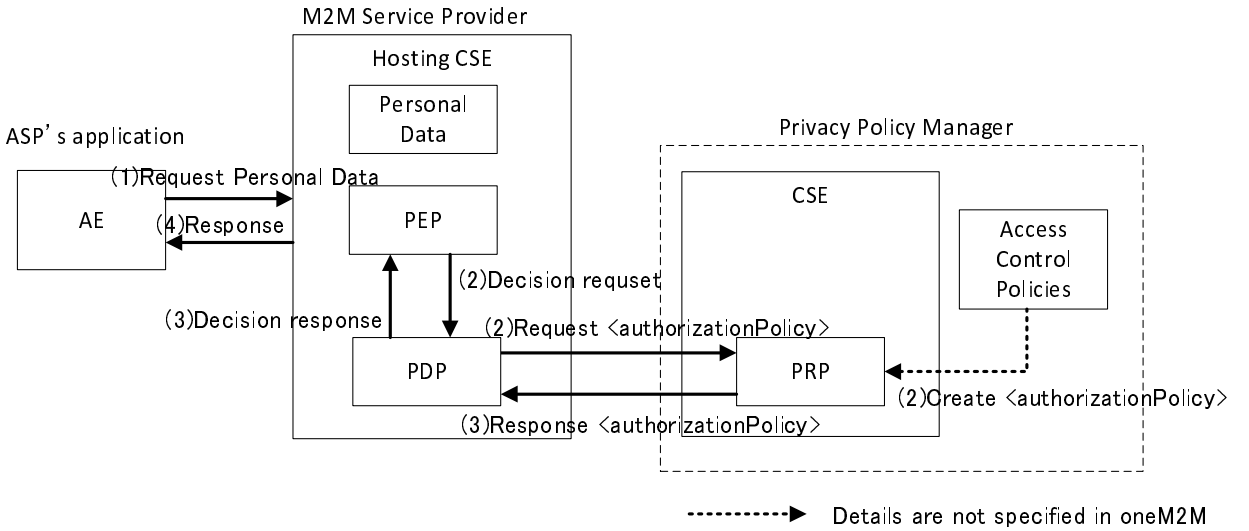


Figure 11.3.3.3.3-1: Request for personal data to the Hosting CSE (the PPM works as PRP)

1. The ASP requests personal data from the Hosting CSE in the M2M Service Provider.
2. PEP in the Hosting CSE requests access control decision from PDP in the Hosting CSE. Then, the PDP requests <*authorizationPolicy*> from the PPM. The PPM shall create <*authorizationPolicy*> using access control information.

   The PPM could use <*accessControlPolicy*> resources as access control policies. In this case, CSE in the PPM stores <*accessControlPolicy*>
3. The PPM respond <*authorizationPolicy*> to PDP in the Hosting CSE. Then, the PDP decides to permit or deny access to the personal data using the <*authorizationPolicy*> and sends a result as "Decision Response" to PEP.
4. If accessing personal data is permitted, the Hosting CSE accesses the personal data and sends the personal data to the ASP as a response.

11.3.3.3.4 Option 3: PPM works as DAS Server

11.3.3.3.4.1 Option 3.1: Direct Dynamic Authorization

For this option, the PPM shall be implemented as AE and shall provide an interface that enables access control for personal data using the PPM as DAS Server.

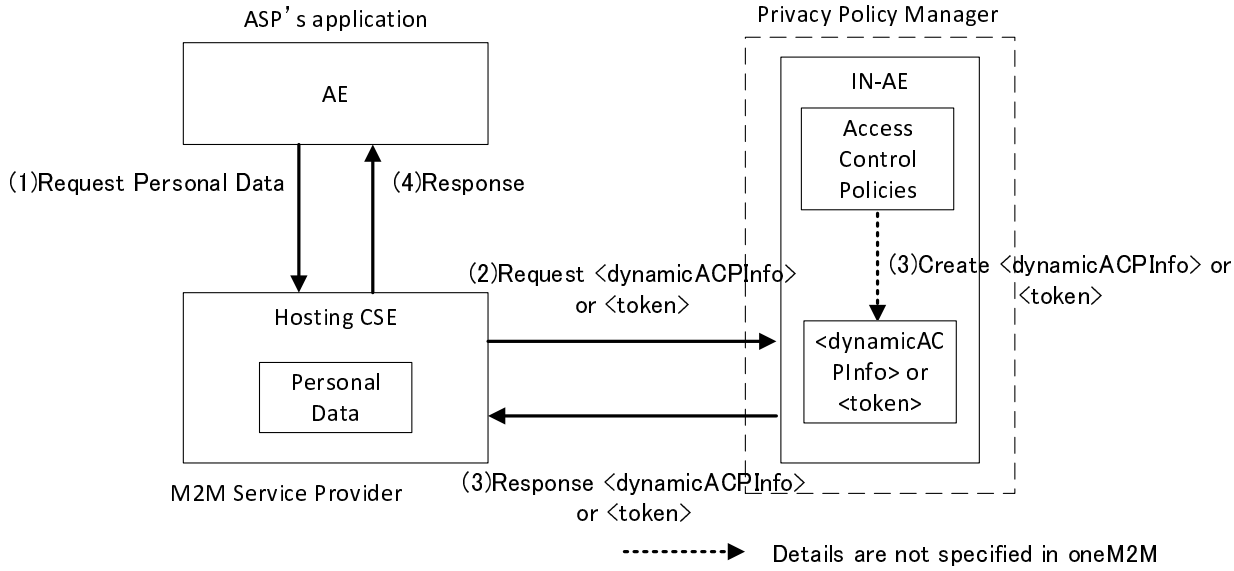


Figure 11.3.3.3.4.1-1: Request for personal data to the Hosting CSE (the PPM works as DAS Server and case of direct dynamic authorization)

1. The ASP requests personal data from the Hosting CSE in the M2M Service Provider.
2. The Hosting CSE performs procedure of direct dynamic authorization. The Hosting CSE checks <*dynamicAuthorizationConsultation*> and requests *dynamicACPInfo* or <*token*> from the PPM.
3. The PPM creates *dynamicACPInfo* or <*token*> based on access control policies and respond *dynamicACPInfo* or <*token*> to the Hosting CSE.

   The PPM could use <*accessControlPolicy*> resources as access control policies.
4. The Hosting CSE make access control decision. If accessing personal data is permitted, the Hosting CSE accesses the personal data and sends the personal data to the ASP as a response.

11.3.3.3.4.2 Option 3.2: Indirect Dynamic Authorization

For this option, the PPM shall be implemented as AE and shall provide an interface that enables access control for personal data using the PPM as DAS Server. In this clause, some optional procedures of indirect dynamic authorization are omitted and focused on procedures related to the PPM.

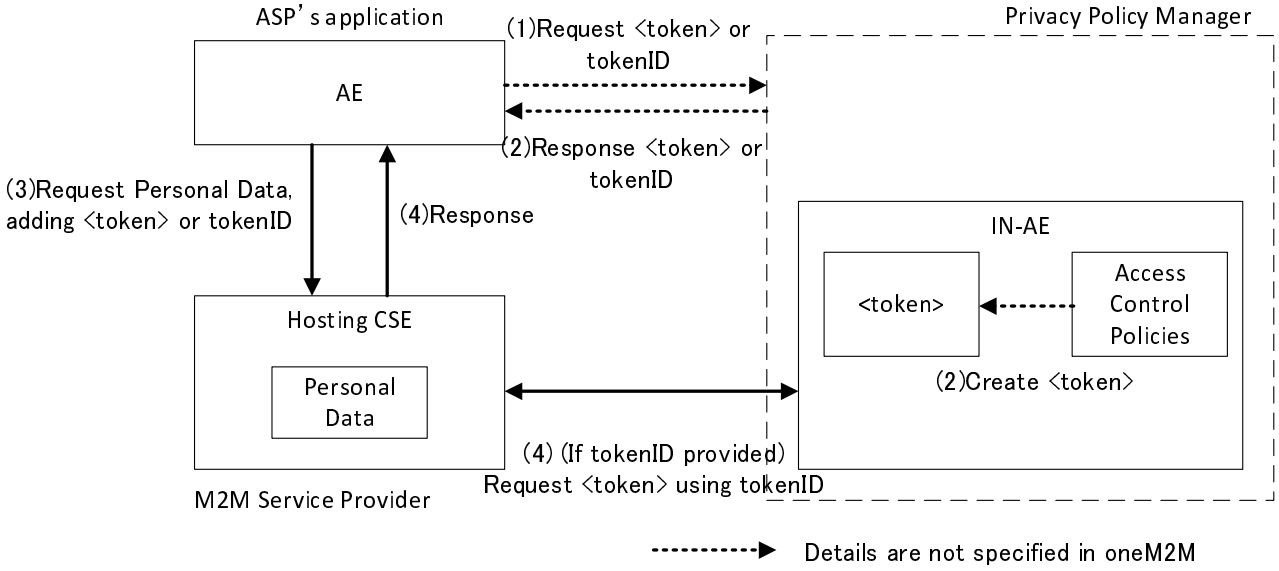


Figure 11.3.3.3.4.2-1: Request for personal data to the Hosting CSE (the PPM works as DAS Server and case of indirect dynamic authorization)

1. The ASP requests <*token*> or tokenID from the PPM.
2. The PPM creates <*token*> and respond <*token*> or tokenID to the ASP.

   Details of above two procedures are not specified in oneM2M.
3. The ASP requests personal data from the Hosting CSE with <*token*> or tokenID.

The Hosting CSE make access control decision using <*token*>. If the Hosting CSE receive tokenID, the Hosting CSE requests <*token*> from the PPM with tokenID. If accessing personal data is permitted, the Hosting CSE accesses the personal data and sends the personal data to the ASP as a response.

11.4 Privacy Policy Manager Implementation Models

11.4.1 Using Terms and Conditions Mark-up Language

11.4.1.0 Introduction

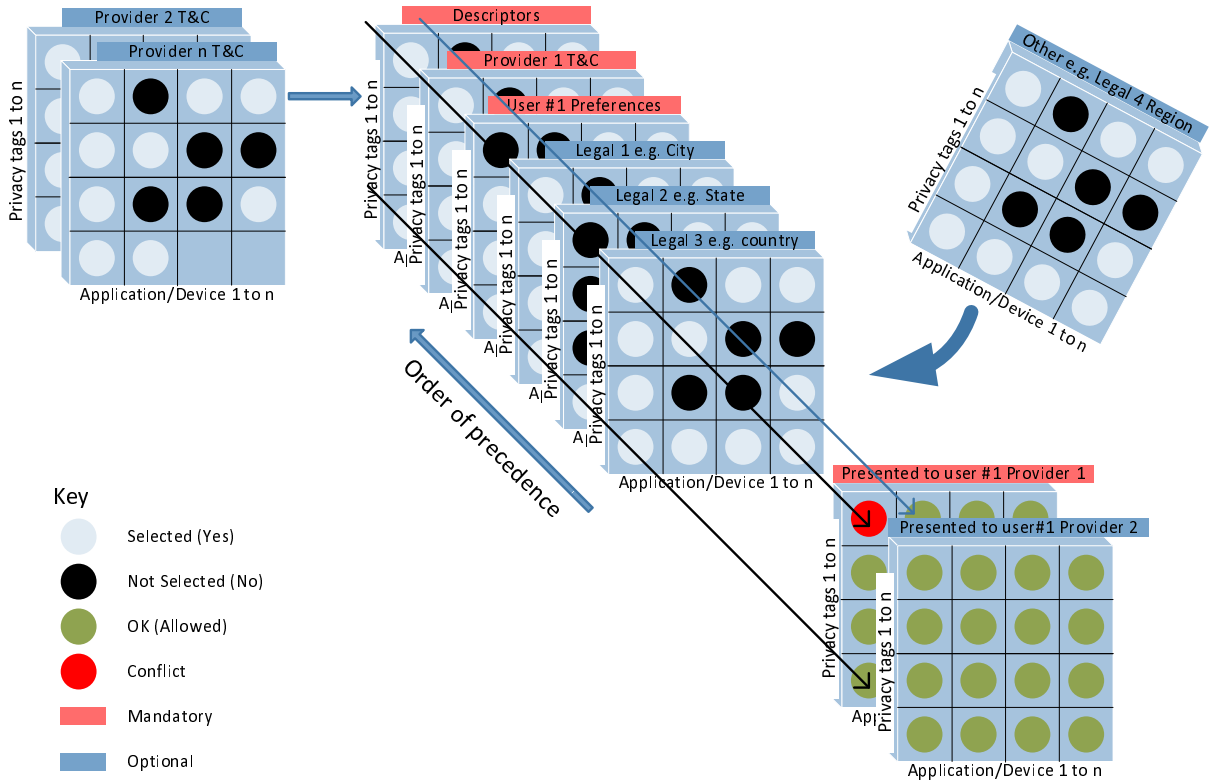


Figure 11.4.1.0-1: Privacy Policy Manager Implementation Model Using Terms and Conditions Mark-up Language, for one end user (#1) and one Application Service Provider (Provider 1)

The above model views the components of the Privacy Policy Manager (PPM) for one end user (#1) and one ASP (Provider 1), arranged as a number of selected/not selected filters in a series of stackable Filter Frames.

Four mandatory Filter Frames are defined:

1) Descriptor Filter Frame.

2) At least one "Provider Terms and Conditions" Filter Frame.

3) User Preferences Filter Frame.

4) At least one "Presented to user" Filter Frame.

Within each Filter Frame, there are grids representing the Privacy Tags in the Mark-up Language, vertically and the applications and/or devices, horizontally.

For the Provider Terms and Condition Filter Frame and User Preferences Filter Frame, each attribute represented by the privacy tag configured as being "selected" or "not selected" for a particular application/device is modelled by "dropping in" an appropriate coloured filter disc.

Discs at the same positions within one or more similarly structured "Presented to user Filter Frames" detect clear paths through the Filter Frame stack:

- Where provider terms and conditions and user preferences are in agreement, these discs turn green.
- Where the paths are blocked by one or more conflicts, similar detectors turn the discs red.

EXAMPLE: If the Application Service Provider expects the user to agree to location information to be collected and shared with a 3rd party, then the ASP selects those two attributes (clear discs) If the end user has set a preference that they do not want location information to be collected and shared, then there will be black discs in the User Preferences Filter Frame and path through the stack will be blocked.

Optional additional Filter Frames may be placed in the stack to "select" or "not select" those same features again by "dropping in" an appropriate coloured filter disc. For example, a country Policy Precedence mandate may overrule an application Service Provider or end user selection. The position of these optional Filter Frames determines the precedence, with those at the front overruling those at the back.

The assumption with this model is that the vast majority of the provider attributes selected by the application Service Provider will not conflict with user preferences and will show green. However, there will be a very large numbers of devices, applications and frequency of software updates, and additions replacements of devices. While most will not result in a conflict, those that do will be instantly identified by one or more red discs which are only displayed to the end user, thus avoiding the need to constantly read and reread hundreds of pages of detailed T&Cs.

There shall be an instance of this stack for each end user who is registered with the PPM and an instance for each Application Service Provider for which they have subscribed. However, the Descriptor Filter Frame and optional city/state/country/region Filter Frames may be shared resources for these instances.

While the description software implementation of this model is outside the scope of the present, document sample code for implementation of the logic is shown in annex K (informative).

11.4.1.1 Registration of Application Service Provider Privacy Policy

1) Optional registration of an applications Privacy Policy shall be part of the process of obtaining a Registered App-ID for each application and version and presenting a security certificate to the oneM2M Registration Authority that is used to authenticate the application and version.

2) The ASP shall download an application Terms and Conditions (T&C) import template from the oneM2M App-ID Registry server, if they do not already have the correct application T&C import template.

3) The application T&C import template shall list in numeric order the tags in normative annex J.

NOTE: The format of the T&C import template is left to implementation, as long as it is able to convey the information specified in annex J.

4) For each tag in the list, the ASP shall provide a value for all devices and applications in the scope of the application that the ASP is registering in the format defined in normative annex J.

5) The ASP shall process the application T&C import template using their local systems and procedures with input from devices vendors and third parties who provide components of their application to create one or more provider T&Cs.

6) The oneM2M App-ID Registry shall, at a minimum, also provide the ASP with the "descriptors list" in the language of the oneM2M partner to support the ASP in completing the T&C import templates to form the set of Provider T&C for that ASP.

7) The security certificate that was used during the App-ID registration process shall also be used to ensure integrity and protect the completed application T&C import template in subsequent storage and transmission.

8) The oneM2M App-ID Registry shall check the authenticity and integrity of the ASP T&Cs by verifying the signature with the ASP public key certificate during App-ID Registration.

9) Each ASP or software vendor T&C completed shall be associated to the App-ID in the oneM2M App-ID Registry.

11.4.1.2 Registration of End User Privacy Preferences

1) When an end user subscribes to a service provided by an application service provider, the end user becomes a data subject, and the data subject downloads or views the end user privacy preferences template from the PPM Portal.

2) The template used by the end user to state their privacy preferences shall align with the template used by the Application Service Provider i.e. the tags as listed in normative annex J shall be displayed in the same order.

3) The end user selects and deselects attributes to state their privacy preferences which are then registered on the PPM using the same portal.

11.4.1.3 Creating a customized Privacy Policy for each end user

1) To make it easy for the data subject to confirm differences between the privacy preference and the privacy policy:

   a) If the ASP's selection of the feature represented by the tag value matches the privacy preference selected by the user for that Application/Device, then the corresponding "presented to user" indicator shall be set to green.

   b) If the ASP's non selection of the feature represented by the tag value matches the privacy preference set by the user for that Application/Device, then the corresponding "presented to user" indicator shall be set to green.

   c) If the ASP's value selected for the feature represented by the tag value matches the privacy preference selected by the user for that Application/Device, then the corresponding "presented to user" indicator shall be set to green.

   d) If the ASP's selection of the feature represented by the tag value does not match the privacy preference selected by the user for that Application/Device, then the corresponding "presented to user" indicator shall be set to red.

   e) If the ASP's non selection of the feature represented by the tag value does not match the privacy preference selected by the user for that Application/Device, then the corresponding "presented to user" indicator shall be set to red.

   f) If the ASP's value set for the feature represented by the tag value does not match the privacy preference set by the user for that Application/Device, then the corresponding "presented to user" indicator shall be set to red.

2) The above rules shall be overridden if one or more optional preference profiles are present.

3) The order of precedence shall be:

   1) Legal Region.

   2) Legal Country.

   3) Legal City.

   4) Legal State.

   5) Parental Control.

12. Security-Specific oneM2M Data Type Definitions

12.1 Introduction

Clause 12 contains data type definitions used only within the oneM2M security specifications.

Any data types of XML elements defined for use only within oneM2M security specifications shall use the namespace:

- https://www.onem2m.org/technical/specifications.

The present document, and any XML or XML Schema Documents produced by oneM2M shall use the prefix "sec:" to refer to that namespace.

12.2 Simple Security-Specific oneM2M Data Types

Table 12.2-1 describes simple data type definitions specific to security. The types in table 12.2-1 are either:

- Atomic data types derived from XML Schema data types by restrictions other than enumeration.
- List data types constructed from other XML Schema or oneM2M-defined atomic data types.

Table 12.2-1: Security-specific oneM2M simple data types

XSD type name	Used for	Examples	Description
sec:relKeyID	Relative part of symmetric key Identifiers	1he83he, my-key_name, firstname.lastname	Any combination of the Roman alphabet, numerals, '.', '_' and '-' characters
sec:credentialID	Credential Identifier	10-thiskey@mymef.com	A sec:credIDTypeID and a xs:anyURI separated by the '-' character. See clause 10.4. The xs:anyURI is the value part of the credential-ID
sec:deviceConfig URI	*deviceConfigURI* attribute of the <*MEFBase*> resource, see ETSI TS 118 132 [58]	1:http://server.dmprovid er.com	A sec:devMgmtID value (see clause 12.3.2.2) separated with colon ":" from the URI of a device management server

12.3 Enumerated Security-Specific oneM2M Data Types

12.3.1 Introduction

The enumerated security-specific oneM2M data types are treated identically to the enumerated oneM2M data types defined in clause 6.3.4 of ETSI TS 118 104 [4]. These data types are based on <xs:integer>, with the numeric values interpreted as specified in clause 12.3.2.

12.3.2 Enumeration type definitions

12.3.2.1 sec:credIDTypeID

The sec:credIDTypeID enumeration type is used in sec:credentialID to identify the type of the identified credential.

Table 12.3.2.1-1: Interpretation of the sec:credIDTypeID enumeration type

Value	Interpretation	Note
10	Symmetric key used to authenticate to a MEF (KpmID)	See clause 8.3.2.1
11	Symmetric key used to authenticate to a MAF (KmID)	See clause 8.8.3.1
12	Symmetric key used to authenticate in an SAEF (KpsaID or KcID)	See clauses 8.2.2.1, 8.2.2.3
13	Symmetric key used to authenticate in ESPrim (pairwiseESPrimKeyID)	See clauses 8.4.2
14	Symmetric key used for direct encryption in the ESData Encryption-only or Nested-Sign-then-encrypt security classes (generic symmetric key identifier format)	See clause 8.5.2
15	Symmetric key used for symmetric key wrap in the ESData Encryption-only or Nested-Sign-then-encrypt security classes (generic symmetric key identifier format)	See clause 8.5.2
16	Symmetric key used for HMAC in the ESData Signature-only security class (generic symmetric key identifier format)	See clause 8.5.2
30	Raw Public Key Certificate used in TLS: (Public Key Identifier)	See clause 10.1.2
31	Device Certificate used in TLS (globally unique hardware instance identifier)	See clause 10.1.1.4
32	CSE-ID Certificate used in TLS (CSE-ID)	ETSI TS 118 101 [1]
33	AE-ID Certificate used in TLS (AE-ID)	ETSI TS 118 101 [1]
34	Node-ID Certificate used in TLS (Node-ID)	See clause 10.1.1.8
41	Raw Public Key Certificate used for RSA or ECDH Key management in the ESData Encryption-only or Nested-Sign-then-encrypt security classes: (Public Key Identifier)	See clause 10.1.2
42	Device Certificate used for RSA or ECDH Key management in the ESData Encryption-only or Nested-Sign-then-encrypt security classes (globally unique hardware instance identifier)	See clause 10.1.1.4
43	CSE-ID Certificate used for RSA or ECDH Key management in the ESData Encryption-only or Nested-Sign-then-encrypt security classes (CSE-ID)	ETSI TS 118 101 [1]
44	AE-ID Certificate used for RSA or ECDH Key management in the ESData Encryption-only or Nested-Sign-then-encrypt security classes (AE-ID)	ETSI TS 118 101 [1]
45	Node-ID Certificate used for RSA or ECDH Key management in the ESData Encryption-only or Nested-Sign-then-encrypt security classes (Node-ID)	See clause 10.1.1.8
51	Raw Public Key Certificate used for RSA or ECDH Key management in the ESData Signature-only security class: (Public Key Identifier)	See clause 10.1.2
52	Device Certificate used for RSA or ECDSA signatures in the ESData Signature-only security class (globally unique hardware instance identifier)	See clause 10.1.1.4
53	CSE-ID Certificate used for RSA or ECDH Key management in the ESData Signature-only security class (CSE-ID)	ETSI TS 118 101 [1]
54	AE-ID Certificate used for RSA or ECDH Key management in the ESData Signature-only security class (AE-ID)	ETSI TS 118 101 [1]
55	Node-ID Certificate used for RSA or ECDH Key management in the ESData Signature-only security class (Node-ID)	See clause 10.1.1.8
NOTE: The form of the identifier for the credential type is described in brackets.		

12.3.2.2 sec:devMgmtID

The sec:devMgmtID enumeration type is used in sec:deviceConfigURI as an identifier of the device management technology used for field device configuration (cf. ETSI TS 118 122 [57]). The sec:devMgmtID enumeration type is also used in the *devMgmtID* element of the *devCfgArgs element* of the *cmdArgs* of the MEF Client Command *cmdDescription* element (see clause 8.3.9.8) to indicate the DM protocol to be used for Device Configuration (ETSI TS 118 122 [57]). The *cmdDescription* is an attribute of the <*mefClientCmd*> resource type in ETSI TS 118 132 [58]).

Table 12.3.2.2-1: Interpretation of the sec:devMgmtID enumeration type

Value	Interpretation	Note
1	OMA DMv1.3	See ETSI TS 118 105 [i.29]
2	OMA DMv2.0	See ETSI TS 118 105 [i.29]
3	OMA LwM2M	See ETSI TS 118 105 [i.29]
4	BBF TR-069	See ETSI TS 118 106 [i.30]

12.3.2.3 sec:cmdClassID

The sec:cmdClassID enumeration type is used in the MEF Client Command *cmdDescription* element (see clause 8.3.9.4) to indicate the *cmdClass* of the *cmdDescription*. The *cmdDescription* is an attribute of the <*mefClientCmd*> resource type specified in ETSI TS 118 132 [58]).

Table 12.3.2.3-1: Interpretation of the sec:cmdClassID enumeration type

Value	Interpretation	Note
0	NO_MORE_COMMANDS	The command class is specified in clause 8.3.9.6.
1	CERT_PROV	The command class is specified in clause 8.3.9.7 Certificate Provisioning is specified in clause 8.3.6
2	DEV_CFG	The command class is specified in clause 8.3.9.8 Device Configuration is specified in ETSI TS 118 122 [57]
3	MO_NODE	The command class is specified in clause 8.3.9.9

12.3.2.4 sec:cmdStatusCode

The sec:cmdStatusCode enumeration type is used by the *cmdStatusCode* element to indicate the status of an MEF Client Command. The *cmdStatus* is an attribute of the <*mefClientCmd*> resource type specified in ETSI TS 118 132 [58].

Table 12.3.2.4-1: Interpretation of the sec:cmdStatusCode enumeration type

Value	Interpretation	Note
10	MEF_CLIENT_CMD_ISSUED	See clause 8.3.9.5.2
11	MEF_CLIENT_CMD_REISSUED	See clause 8.3.9.5.3
20	MEF_CLIENT_CMD_OK	See clause 8.3.9.5.4
40	MEF_CLIENT_CMD_REPEATED_CMD_ID	See clause 8.3.9.5.5
41	MEF_CLIENT_CMD_CLASS_NOT_SUPPORTED	See clause 8.3.9.5.6
42	MEF_CLIENT_CMD_BAD_ARGUMENTS	See clause 8.3.9.5.7
43	MEF_CLIENT_CMD_UNACCEPTABLE_ARGUMENTS	See clause 8.3.9.5.8
100	MEF_CLIENT_CMD_CERT_PROV_SERVER_ERROR	See clause 8.3.9.5.9
101	MEF_CLIENT_CMD_CERT_PROV_CLIENT_ERROR	See clause 8.3.9.5.10
201	MEF_CLIENT_CMD_DEV_CFG_SERVER_ERROR	See clause 8.3.9.5.11
202	MEF_CLIENT_CMD_DEV_CFG_CLIENT_ERROR	See clause 8.3.9.5.12
300	MEF_CLIENT_CMD_MO_NODE_NOT_FOUND	See clause 8.3.9.5.13
301	MEF_CLIENT_CMD_MO_NODE_TYPE_CONFLICT	See clause 8.3.9.5.14
302	MEF_CLIENT_CMD_MO_NODE_BAD_ARGS	See clause 8.3.9.5.15
303	MEF_CLIENT_CMD_MO_NODE_UNACCEPTABLE_ARGS	See clause 8.3.9.5.16
304	MEF_CLIENT_CMD_MO_NODE_INCONSISTENT_CONFIG	See clause 8.3.9.5.17
305	MEF_CLIENT_CMD_MO_NODE_EXECUTION_ERROR	See clause 8.3.9.5.18

12.3.2.5 sec:certProvProtocolID

The sec:certProvProtocolID enumeration type is used for the *certProvProtocolID* element of the *certProvCmdArgs* element of the *cmdArgs* element of the MEF Client Command *cmdDescription* element (see clause 8.3.9.7) to indicate the Certificate Provisioning protocol to be used. The *cmdDescription* is an attribute of the <*mefClientCmd*> resource type specified in ETSI TS 118 132 [58]).

Table 12.3.2.5-1: Interpretation of the sec:certProvProtocolID enumeration type

Value	Interpretation	Note
1	EST	See clause 8.3.6.2
2	SCEP	See clause 8.3.6.3

12.3.2.6 sec:certSubjectType

The sec:certSubjectType enumeration type is used for the *certSubjectType* element of the *certProvCmdArgs* element of the *cmdArgs* element of the MEF Client Command *cmdDescription* element (see clause 8.3.9.7) to indicate if the subject of the provisioned certificate will be a Node, CSE or AE. The *cmdDescription* is an attribute of the <*mefClientCmd*> resource type specified in ETSI TS 118 132 [58]).

Table 12.3.2.6-1: Interpretation of the sec:certSubjectType enumeration type

Value	Interpretation	Note
1	Node-ID	See ETSI TS 118 101 [1], clause 7.1.5
2	CSE-ID	See ETSI TS 118 101 [1], clause 7.2
3	AE-ID	See ETSI TS 118 101 [1], clause 7.2

12.3.2.7 sec:objectTypeID

The sec:objectTypeID enumeration type is used for the *objectTypeID* element of the *MONodeCmdArgs* element of the *cmdArgs* element of the MEF Client Command *cmdDescription* element (see clause 8.3.9.9) to indicate the type of an MO Node. The *cmdDescription* is an attribute of the <*mefClientCmd*> resource type specified in ETSI TS 118 132 [58]).

Table 12.3.2.7-1: Interpretation of the sec:objectTypeID enumeration type

Value	Interpretation	Note
1	[*authenticationProfile*]	See ETSI TS 118 122 [57], clause 7.1.4 and 7.2.4.
2	[*trustAnchorCred*]	See ETSI TS 118 122 [57], clause 7.1.6 and 7.2.6.
3	[*MAFClientRefCfg*]	See ETSI TS 118 122 [57], clause 7.1.7 and 7.2.7.

12.4 Complex Security-Specific oneM2M Data Types

12.4.1 MAF and MEF client configuration data

Table 12.4.1-1 defines the assignment of data types to the four data containers which are used in MAF and MEF client registration and key registration configuration procedures. Note that these data containers are not defined in the form of resource types since the information is not remotely accessible. The information elements of these containers are managed by means of Device Management procedures (see ETSI TS 118 122 [57]) or by manual provisioning.

Table 12.4.1-1: Types used in MAF and MEF Registration Configuration procedures

data container name	Used in	Data Type	Notes
mefClientRegCfg	MEF Client Registration Configuration, see clause 8.3.7.2	sec:clientRegCfg	See clause 12.4.2

data container name	Used in	Data Type	Notes
mafClientRegCfg	MAF Client Registration Configuration, see clause 8.8.3.2		
mefKeyRegCfg	MEF Key Registration Configuration, see clause 8.3.7.3	sec:keyRegCfg	See clause 12.4.3
mafKeyRegCfg	MAF Key Registration Configuration, see clause 8.8.3.3		

12.4.2 sec:clientRegCfg

Data type sec:clientRegCfg applies to the *mefClientRegCfg* and *mafClientRegCfg* data containers used in MEF Client Registration Configuration and MAF Client Registration Configuration, see clause 8.3.7.2 and clause 8.8.3.2, respectively.

Table 12.4.2-1: Type definition of sec:clientRegCfg

Element Path	Element Type	Multiplicity	Notes
expirationTime	m2m:timestamp	0..1	See ETSI TS 118 104 [4]
labels	m2m:labels	0..1	See ETSI TS 118 104 [4]
fqdn	xs:anyURI	1	
adminFQDN	xs:anyURI	1	
httpPort	xs:unsignedByte	0..1	
coapPort	xs:unsignedByte	0..1	
websocketPort	xs:unsignedByte	0..1	

12.4.3 sec:keyRegCfg

Data type sec:keyRegCfg applies to the *mefKeyRegCfg* and *mafKeyRegCfg* data containers used in MEF Key Registration Configuration and MAF Key Registration Configuration, see clause 8.3.7.3 and clause 8.8.3.3, respectively.

Table 12.4.3-1: Type definition of sec:keyRegCfg

Element Path	Element Type	Multiplicity	Notes
expirationTime	m2m:timestamp	0..1	See ETSI TS 118 104 [4]
labels	m2m:labels	0..1	See ETSI TS 118 104 [4]
adminFQDN	xs:anyURI	1	
SUID	m2m:suid	1	See ETSI TS 118 104 [4]
targetIDs	m2m:listOfM2MID	0..1	See ETSI TS 118 104 [4]

12.4.4 sec:cmdDescription

The sec:cmdDescription complex type is used by the *cmdDescription* element to describe an MEF Client Command, described in clause 8.3.9.5. The *cmdDescription* is an attribute of the <*mefClientCmd*> resource type specified in ETSI TS 118 132 [58].

Table 12.4.4-1: Type definition of sec:cmdDescription

Element Path	Element Type	Multiplicity	Notes
cmdClassID	sec:cmdClassID	1	See clause 12.3.2.3
cmdArgs	sec:cmdArgs	1	See clause 12.4.5
targetID	m2m:ID	1	ETSI TS 118 104 [4]

12.4.5 sec:cmdArgs

The sec:cmdArgs complex type is used by the *cmdArgs* element of datatype sec:cmdDescription.

Table 12.4.5-1: Type definition of sec:cmdArgs

Element Path	Element Type	Multiplicity	Notes
noMoreCmdArgs	sec:noMoreCmdArgs	0..1	See clause 12.4.6
certProvCmdArgs	sec:certProvCmdArgs	0..1	See clause 12.4.7
devCfgCmdArgs	sec:devCfgCmdArgs	0..1	See clause 12.4.8
MONodeCmdArgs	sec:MONodeCmdArgs	0..1	See clause 12.4.9

This type is an xs:choice. It shall contain elements from no more than one row listed in table 12.4.5-1.

12.4.6 sec:noMoreCmdArgs

The sec:noMoreCmdArgs complex type is used in sec:cmdDescription.

Table 12.4.6-1: Type definition of sec: noMoreCmdArgs

Element Path	Element Type	Multiplicity	Notes
retryDuration	xs:duration	1	

12.4.7 sec:certProvCmdArgs

The sec:certProvCmdArgs complex type is used in sec:cmdDescription.

Table 12.4.7-1: Type definition of sec:certProvCmdArgs

Element Path	Element Type	Multiplicity	Notes
certProvProtocolID	sec:certProvProtocolID	1	See clause 12.3.2.5
URI	xs:anyURI	1	
certSubjectType	sec:certSubjectType	1	See clause 12.3.2.6
certSubjectID	xs:union of m2m:nodeID and m2m:ID	1	See ETSI TS 118 104 [4], clause 6.3.3.

12.4.8 sec:devCfgCmdArgs

The sec:devCfgCmdArgs complex type is used in sec:cmdDescription.

Table 12.4.8-1: Type definition of sec:devCfgCmdArgs

Element Path	Element Type	Multiplicity	Notes
deviceConfigURI	sec:deviceConfigURI	1	See clause 12.2

12.4.9 sec:MONodeCmdArgs

The sec:MONodeCmdArgs complex type is used in sec:cmdDescription.

Table 12.4.9-1: Type definition of sec:MONodeCmdArgs

Element Path	Element Type	Multiplicity	Notes
objectPath	xs:anyURI	1	
objectTypeID	sec:objectTypeID	1	See clause 12.3.2.7
objectTypeSpecifcArgs	sec:authProfileMONodeArgs	0..1	See clause 12.4.10

12.4.10 sec:authProfileMONodeArgs

The sec:authProfileMONodeArgs complex type is used in sec:MONodeCmdArgs.

Table 12.4.10-1: Type definition of sec:authProfileMONodeArgs

Element Path	Element Type	Multiplicity	Notes
SUID	m2m:suid	1	See ETSI TS 118 104 [4], clause 6.3.4.2.39

Annex A (informative): Mapping of 3GPP GBA terminology

Table A-1 provides a mapping of terminology and abbreviations used in GBA according to 3GPP specification [13] to corresponding oneM2M terminology and abbreviations as used within the present document.

Table A-1

GBA entities, keys and processes	oneM2M Security Bootstrap entities, keys & processes
UE	Enrolee
BSF	MEF
NAF	MAF
Bootstrapping Procedure	**Bootstrap Security Handshake + Temporary Enrolment Key Generation**
Ks	Ke
B-TID	KeID
Bootstrapping Usage Procedure	**Usage in MAF Handshake**
NAF FQDN	MAF-ID
Ks_(ext/int)_NAF	Km (Master Credential)

Annex B (informative): General Mutual Authentication Mechanism

B.0 Introduction

oneM2M mutual authentication schemes allow oneM2M entities to prove that they know related credentials such as Master Credentials, without having to exchange value of those credentials, and sensitive data such as security identities and security identifiers. To prevent reading and copying of credentials, a secure environment within the Security CSF provides protection against tampering of those credentials and related processed information.

A general mutual authentication protocol is applied to both symmetric and asymmetric key based schemes. Precise protocol messages and parameters depend on the chosen scheme and the security parameters selected. Typically it consists of following steps as shown in figure B.0-1.

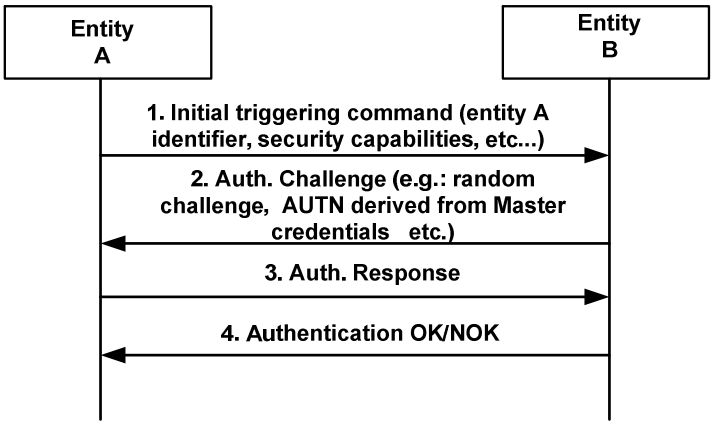


Figure B.0-1: Mutual Authentication

1. An initial step where an entity A is securely identified to an entity B with whom previous or no previous contact has been made. In this step entity A identifies itself to an entity B protected against eavesdropping, i.e. no exchange of key materials (Master Credentials).

2. In the second step entity B sends a challenge to entity A. The Authentication Challenge consists of a challenge, the authentication token (AUTN) of entity B derived from Master Credentials, etc. The authentication challenge, which may be random or not, depends on the chosen authentication scheme and the security parameters selected for symmetric and asymmetric key based schemes.

3. Entity A replies with an Authentication Response that contains an authentication token (AUTN) derived from its known Master Credentials and the received Authentication Challenge. This Authentication Response is sent if entity B has been successfully authenticated by entity A.

4. Entity B then verifies the relation between entity A's identity and the response received in step 3. If the verification is positive, entity B is assured that the response has been created by entity A using a secret associated with entity A's identity provided in step 1.

B.1 Group Authentication

The oneM2M transactions may naturally involve groups of M2M entities rather than individual ones. A number of entities are classified as a group due to their proximate locations, having the same features, belonging to the same owner, or any other reasons. To get services, all entities in such a group should be authenticated first. The traditional authentication mechanism has two main solutions, the first authentication mechanism is that the service provider authenticates each entity in the group one by one; the second authentication mechanism is that each entity makes mutual authentication with a group agent, then the group agent makes mutual authentication with the service provider. If the first authentication mechanism is used, the resulting authentication overheads of computation and communication may be too high to afford. If the second authentication mechanism is used, it has the following security weaknesses:

a) It may exist the man-in-the-middle attack by the group agent: The group agent would be placed in unsecure place or owned by different provider rather than the service provider. If the group agent is compromised or lie to service provider, group agent would act as a middle attacker to make fake authentication to entities and report fake identity to service provider since there is no direct authentication from service provider to each M2M entity.

b) Privacy concern: All information from M2M entities is transferred through the group agent, and the group agent knows all information generated by each entity. Based on security consideration, if the group agent is owned by different owner other than the entities' and service providers' owner, the group agent should not get the message.

Hence, the M2M entities (e.g. ASN or ADN) with the same feature can utilize group authentication to service provider (e.g. infrastructure node) in order to provide end-to-end secure tunnel as well as reducing the communication overhead.

Annex C (normative): Security protocols associated to specific SE technologies

C.0 Introduction

The Secure Environment supporting security functions specified by oneM2M provides a level and a type of protection (e.g. integrity protection, confidentiality, tamper resistance) to the information it contains, independently of the method of protection (e.g. UICC, embedded security element, TEE, etc.). Administration of their content is implementation dependent and relies on existing standards within specific Secure Environment technologies. Some of them are listed below for information.

C.1 UICC

In case of UICC (SE compliant with ETSI TS 102 671 [23]), OTA mechanisms as specified in [7] and [8], and its extensions [9], [10] for 3GPP underlying networks or [11] and [12] for 3GPP2 underlying networks shall be supported to enable security administration of the sensitive data of the M2M Service Layer. UICC provides the highest protection level 3 against attacks according the Classification of Protection levels table 6.3.1-1 in clause 6.3.1.

C.2 Other secure element and embedded secure element with ISO/IEC 7816 interface

In case the Secure Environment is implemented as a security element or as an embedded security element supporting an ISO/IEC 7816-4 interface [26], example of remote administration can be according to GlobalPlatform Remote Administration [47]. An embedded secure element provides the highest protection level 3 against attacks according the Classification of Protection levels table 6.3.1-1 in clause 6.3.1.

C.3 Trusted Execution Environment

In case the secure environment is implemented as a Trusted Execution Environment (TEE) according to GlobalPlatform [22], remote administration shall be supported as specified in GlobalPlatform Remote Administration [21]. TEE provides the medium protection level 2 against attacks according the Classification of Protection levels table 6.3.1-1 in clause 6.3.1.

C.4 SE to CSE binding

In case the SE is implemented as an independent security element supporting ETSI TS 102 221 [24], the platform-to-platform secure channel specified in ETSI TS 102 484 [25] provides logical binding of the SE to a specific CSE or AE. This also protects the information exchanged between the SE and the associated entity on physically exposed interfaces, and is therefore recommended for devices that are physically exposed to attackers.

Annex D (normative): UICC security framework to support symmetric key based oneM2M Services

D.0 Introduction

This annex is applicable when UICC (a type of Independent Secure Element compliant with ETSI TS 102 221 [24] and ETSI TS 102 671 [23]) is involved in M2M service layer security using Pre-Shared symmetric Keys, whether it only serves as a mean to pre-provision M2M Service layer material in M2M Devices/Gateways, or it is further used as Secured Environment in an M2M Device/Gateway.

In case of M2M deployments using asymmetric credentials (e.g. Public Key cryptography), or in case support of advanced UICC features such as File System support is not desired, annex L of the present document provides an interoperable framework that can be implemented on tamper resistant hardware secure elements without relying on UICC specific features.

Specifically, the involvement of UICC in oneM2M security may include any of the following steps:

- Pre-provisioning of initial PSK credentials in M2M nodes by any of the following methods:
  - Simple pre-provisioning and administration of M2M Service material (initial credentials and other pre-provisioned parameters), i.e. UICC-based M2M service provisioning;
  - Support for infrastructure assisted bootstrapping of the M2M symmetric credentials by derivation from symmetric Access Network credentials stored in the UICC, using GBA.
- Derivation of a security association key directly derived from symmetric Access Network Credentials, using GBA. Note that this process can be supported by a Network Access Application on the UICC independently of the presence of the information structure specified in the present annex.

The support of UICC provisioning of M2M service subscription information shall be indicated in the M2M Service Table for the corresponding M2M Service Subscription as specified in the present annex.

The support of key derivation using GBA that may be used for bootstrapping or security association shall always be indicated in the Service Table of the UICC application of the Access Network Operator supporting the GBA infrastructure.

At the most basic level, UICC-based M2M pre-provisioning requires an interoperable framework to store and administrate related information in the UICC. Further involvement requires a framework for discovery of available services offered by the UICC for the hosting M2M field node. The purpose of the present annex is to specify this framework, which enables both initial service provisioning and remote security administration of the subscription information during the subscription lifetime.

A common scenario is where an M2M field node holds a UICC application protecting Access Network security credentials, and these credentials are used to derive M2M Service Layer security credentials used for M2M service bootstrapping or security association establishment in the service layer. As these scenarios require a trust agreement between the involved Access Network operator and M2M Service Provider, UICC support for M2M services in such situation shall be handled within the context of the associated Network Access application on the UICC. In particular, the UICC support for M2M credentials derivation using GBA shall be indicated within the UICC application of the Access Network operator. This is specified in clause D.1.

Even when the M2M Service Layer credentials are not derived from Access Network Credentials, the UICC may be used as a secure environment that securely protects the symmetric credential used to root security in an M2M field node. In such cases, the M2M subscription information and related methods constitute an independent application that resides on a UICC, in the sense of ETSI TS 102 221 [24]. In particular, ETSI TS 102 221 [24] specifies the application independent properties of the UICC/terminal interface such as the physical characteristics and the logical structure.

NOTE: A terminal in the sense of ETSI TS 102 221 [24] is the part of the M2M field node that holds the UICC, e.g. a communication modem or an M2M Node processing environment.

The specific properties of the M2M Service Provider Identity Module application holding symmetric credentials is specified in clause D.2.

The storage of M2M information elements in the UICC and the procedures used for communication between the hosting M2M field node and the UICC shall be as specified in the present annex. The present annex uses abbreviations and coding conventions defined in ETSI TS 102 221 [24].

D.1 Access Network UICC-based oneM2M Service Framework

D.1.1 Access Network UICC-based oneM2M Service Framework characteristics

An Access Network UICC-based oneM2M Service Framework is always associated with a single M2M Service Subscription and consists of a single DF, DF_{1M2M}, complying with the specifications in clause D.1.3, implemented in the ADF of a Network Access Application on the UICC. This situation addresses the case where a trust relationship has been established between the M2M SP and the AN operator owning the hosting ADF.

NOTE 1: This does not necessarily imply that the Access Network credentials of the corresponding ADF are used to derive the M2M Service Layer Credentials: e.g. an Access Network operator may refuse derivation from Access Network credentials to an M2M Service Provider, but may still accept to provide space on its UICC to pre-provision independent credentials or support service infrastructure-assisted bootstrapping.

There may be several oneM2M service frameworks (DF_{1M2M}) within the ADF of a single Access Network subscription, in case this Access Network subscription is used by several independent M2M Service subscriptions. The file IDs of the DF_{1M2M} in any ADF shall be listed under the corresponding entry in EF_{DIR} as specified in clause D.1.2.

NOTE 2: A single M2M service layer subscription can also use multiple access networks: such subscriptions are best provisioned in a dedicated ADF as specified in clause D.2.

The content of any DF_{1M2M} in an Access Network application ADF shall be as specified in clause D.1.3.

D.1.2 M2M Service Framework discovery for Access Network UICC

When a UICC Network Access application supports one or more M2M Service subscriptions, with a DF_{1M2M}, the EF_{DIR} entry corresponding to this UICC Network Access Application shall contain the following M2M related Data Objects:

- oneM2M Service Framework DO: defining the association between the identifier of one M2M Service Subscription provisioned in the ADF and the related DF corresponding to this M2M subscription. Likewise, each M2M Service Subscription is associated to one DF. Each of these DFs is hereafter referred as DF_{1M2M}.

There shall be as many oneM2M Service Framework Data Objects as there are M2M Service Subscriptions provisioned in the ADF.

Table D.1.2-1: Coding of oneM2M related DOs

Bytes	Length	Description	Status
1	1	Discretionary template tag = '73'	M
2	1	Length of the discretionary template = X	M
3 to (2+X)	X	Discretionary Template	X

Table D.1.2-2: Coding of oneM2M Discretionary Template related DOs

Bytes	Length	Description	Status
1	1	oneM2M service specific data content tag = 'A2'	M
2	1	M2M service specific data content length = Y	M
3 to (2+Y)	Y	M2M service specific data content	M

Table D.1.2-3: Coding of oneM2M Service Specific Data Content related DOs

Bytes	Length	Description	Status
1	1	oneM2M supported service provisioning tag = '80'	M
2	1	Length of the M2M supported service provisioning tag = A	M
3 to 4	2	M2M Dedicated File Identifier for following M2M service subscription	M
5 to (A+2)	(A-2)	M2M Subscription Identifier	M

Coding:

- M2M Dedicated File identifier:
  - Contain the file identifier of the DF_{1M2M} associated to the provisioning of the M2M Service subscription identified in the DO.
- M2M Subscription Identifier:
  - The identifier of the M2M service subscription provisioned in the DF_{1M2M} indicated in the Data Object, encoded in binary format.

D.1.3 Content of files at the DF_{1M2M} level

D.1.3.0 Introduction

This clause specifies the EFs for the M2M service provisioning specific to a single M2M service provider, defining access conditions, data items and coding. A data item is a part of an EF which represents a complete logical entity.

The file structure for DF_{1M2M} is illustrated in figure D.1.3.0-1.

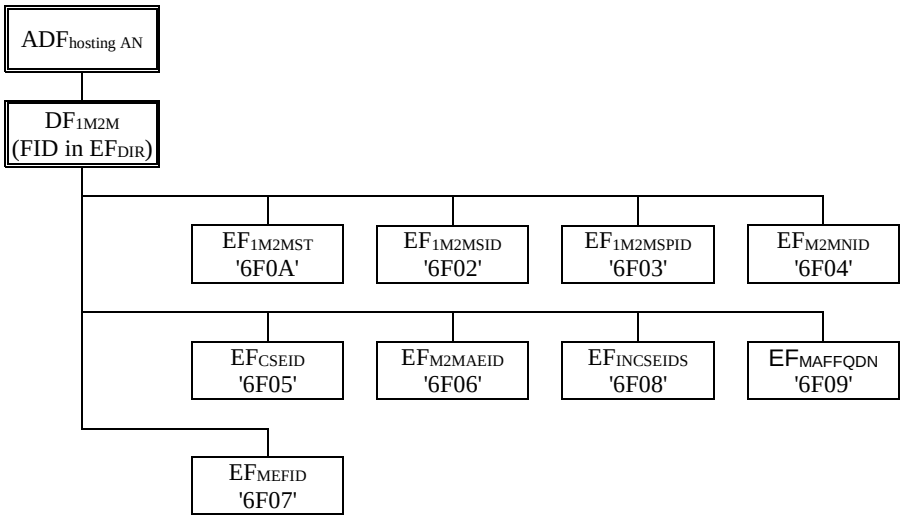


Figure D.1.3.0-1: File identifiers and directory structures of DF_{1M2M} in an hosting Access Network application ADF

D.1.3.1 EF_{1M2MST} (oneM2M Service Table)

This EF indicates which optional oneM2M services are available for the corresponding subscription. If a service is not indicated as available in the oneM2M DF, the hosting M2M field node shall not select this service. The presence of this file is mandatory if optional services are provided by the subscription.

Identifier: '6F0A'	Structure: transparent	Mandatory
SFI: '0A'		
File size: X bytes, X ≥ 1	Update activity: low	

Access Conditions:

READ	ALW
UPDATE	ADM
DEACTIVATE	ADM
ACTIVATE	ADM

Bytes	Description	M/O	Length
1	Services n°1 to n°8	M	1 byte
2	Services n°9 to n°16	O	1 byte
3	Services n°17 to n°24	O	1 byte
4	Services n°25 to n°32	O	1 byte
etc.			
X	Services n°(8X-7) to n°(8X)	O	1 byte

-Services

Contents:

Service n°1:	Local CSE-ID provisioning
Service n°2	IN-CSE-ID list provisioning
Service n°3	MAF FQDN provisioning
Service n°4	Local M2M AE-ID list provisioning
Service n°5	Bootstrapping: MEF address provisioning
Service n°6	M2M-Node-ID information
Service n°7	GBA Secure Provisioning (see note)
Service n°8	GBA Secure Connection (see note)

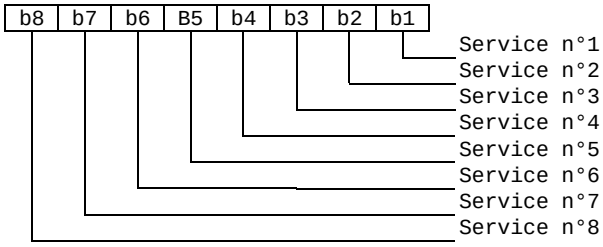
NOTE: Services n°7 and 8 can only be available in a oneM2M Service Table located in a DF_{1M2M} hosted in the ADF of the Network Access Application from which the M2M Service Layer credentials are expected to be derived.

The EF shall contain at least one byte. Further bytes may be included, but if the EF includes an optional byte, then it is mandatory for the EF to also contain all bytes before that byte. Other services are possible in the future and will be coded on further bytes in the EF. Coding:

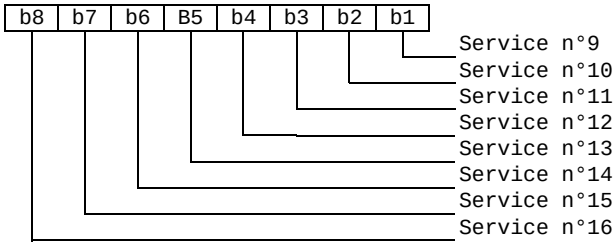
1 bit is used to code each service:
bit = 1: service available;
bit = 0: service not available.

- Service available means that the M2M Service Subscription provisioned in the current DF or ADF has the capability to support the service and that the service is available for the user of the M2M Service Subscription.
  Service not available means that the service shall not be used by the M2M Service Subscription user, even if the M2M Service Subscription has the capability to support the service.

First byte:



Second byte:



etc.

D.1.3.2 $EF_{1M2MSID}$ (oneM2M Subscription Identifier)

This EF contains the oneM2M Subscription Identifier, M2M-Sub-ID. There shall be only one TLV object within this EF.

Identifier: '6F02'	Structure: transparent		Mandatory
SFI: '02'			
File size: X bytes	Update activity: low		
Access Conditions: READ ALW UPDATE ADM DEACTIVATE ADM ACTIVATE ADM			
Bytes	Description	M/O	Length
1	M2M Subscription Identifier TLV data object	M	X bytes

The M2M Subscription Identifier value field shall contain the M2M-Sub-ID encoded as specified in ETSI TS 118 104 [4]. The tag value of the oneM2M Subscription Identifier TLV data object shall be '80'.

D.1.3.3 $EF_{1M2MSPID}$ (oneM2M Service Provider Identifier)

This EF contains the oneM2M Service Provider Identifier, M2M-SP-ID, of the M2M Service Provider related to the subscription in $EF_{1M2MSID}$. There shall be only one TLV object within this EF.

Identifier: '6F03'	Structure: transparent		Mandatory
SFI: '03'			
File size: X bytes	Update activity: low		
Access Conditions: READ ALW UPDATE ADM DEACTIVATE ADM ACTIVATE ADM			
Bytes	Description	M/O	Length
1	M2M-SP-ID TLV data object	M	X bytes

The M2M-SP-ID Value field shall contain the M2M-SP-ID encoded as specified in ETSI TS 118 104 [4]. The tag value of the M2M-SP-ID TLV data object shall be '80'.

D.1.3.4 EF_{M2MNID} (M2M Node Identifier)

This EF contains the M2M-Node-ID supporting the local CSE. It may be used to logically bind a UICC to a specific M2M Node. If service n°6 is "available", this file shall be present. There shall be only one TLV object within this EF.

Identifier: '6F04'	Structure: transparent		Optional
SFI: '04'			
File size: X bytes		Update activity: low	
Access Conditions: READ ALW UPDATE ADM DEACTIVATE ADM ACTIVATE ADM			
Bytes	Description	M/O	Length
1 to X	M2M-Node-ID TLV object	M	X bytes

The M2M-Node-ID Value field shall contain the M2M-Node-ID encoded as specified in ETSI TS 118 104 [4].

D.1.3.5 EF_{CSEID} (local CSE Identifier)

This EF contains the local CSE Identifier, CSE-ID, for the M2M field node associated to the subscription in $EF_{1M2MSID}$. If present, this file is used by the M2M field node to pre-provision the CSE-ID. If service n°1 is "available", this file shall be present. There shall be only one TLV object within this EF.

Identifier: '6F05'	Structure: transparent		Optional
SFI: '05'			
File size: X bytes		Update activity: low	
Access Conditions: READ ALW UPDATE ADM DEACTIVATE ADM ACTIVATE ADM			
Bytes	Description	M/O	Length
1	CSE-ID TLV data object	M	X bytes

CSE-ID TLV

Contents:

- The CSE-ID Value field shall contain the local CSE-ID formatted as a URI.

Coding:

- The URI shall be encoded to an octet string according to UTF-8 encoding rules as specified in IETF RFC 3629 [19]. The tag value of the URI TLV data object shall be '80'.

D.1.3.6 $EF_{M2MAE\text{-}ID}$ (M2M Application Identifiers list)

This EF contains the list of M2M Application Identifiers (AE-IDs) for the local M2M applications supported by the subscription in $EF_{1M2MSID}$. If service n°4 is "available", this file shall be present.

Identifier: '6F06'	Structure: Linear fixed		Optional
SFI: '06'			
Record length: X bytes		Update activity: low	
Access Conditions: READ ALW UPDATE ADM DEACTIVATE ADM ACTIVATE ADM			
Bytes	Description	M/O	Length
1 to X	M2M AE-ID LV data object	M	X bytes

M2M AE-ID LV

Contents:

- The Value field shall contain the M2M AE-ID formatted as a URI.

Coding:

- The URI shall be encoded to an octet string according to UTF-8 encoding rules as specified in IETF RFC 3629 [19].

D.1.3.7 $EF_{INCSEIDS}$ (M2M IN-CSE IDs list)

This EF contains a list of pre-provisioned IN-CSE-ID used to determine the next point of contact after provisioning or M2M Service Bootstrapping. If service n°2 is "available", this file shall be present.

Identifier: '6F08'	Structure: Linear fixed	Optional
Record length: X bytes	Update activity: low	
Access Conditions: READ ALW UPDATE ADM DEACTIVATE ADM ACTIVATE ADM		

Bytes	Description	M/O	Length
1 to X	IN-CSE-ID LV data object	M	X bytes

IN-CSE-ID LV

Contents:

- The Value field shall contain the IN-CSE-ID formatted as a URI.

Coding:

- The URI shall be encoded to an octet string according to UTF-8 encoding rules as specified in IETF RFC 3629 [19].

D.1.3.8 $EF_{MAFFQDN}$ (MAF-FQDN)

This EF is used to pre-provision the FQDN of the MAF to be used for M2M Service Connection after M2M Service Bootstrapping. If service n°3 is "available", this file shall be present. There shall be only one TLV object within this EF.

Identifier: '6F09'	Structure: Transparent	Optional
Length: X bytes	Update activity: low	
Access Conditions: READ ALW UPDATE ADM DEACTIVATE ADM ACTIVATE ADM		

Bytes	Description	M/O	Length
1	MAF FQDN TLV data object	M	X bytes

MAF FQDN

Contents:

- The FQDN address of the MAF.

Coding:

- The MAF-FQDN shall be encoded to an octet string according to UTF-8 encoding rules as specified in IETF RFC 3629 [19]. The tag value of the MAF FQDN TLV data object shall be '80'.

D.1.3.9 EF_{MEFID} (M2M Enrolment Function Identifier)

This EF contains one or more M2M Enrolment Function addresses. The first record in the EF shall be considered to be of the highest priority. The last record in the EF shall be considered to be the lowest priority. If service n°5 is "available", this file shall be present.

Identifier: '6F07'	Structure: linear fixed	Optional
Record length: X bytes	Update activity: low	
Access Conditions: READ ALW UPDATE ADM DEACTIVATE ADM ACTIVATE ADM		

Bytes	Description	M/O	Length
1 to X	MEF Address LV data object	M	X bytes

MEF Address LV data object

Contents:

- Address of MEF, in the format of a FQDN, an IPv4 address, or an IPv6 address.

Coding:

- The format of the data object is as follows:

Field	Length (bytes)
Length	1
Address Type	1
MEF Address	Address Length

- Address Type: Type of the MEF address.
  - This field shall be set to the type of the MEF address according to the following:

Value	Name
0x00	FQDN
0x01	IPv4
0x02	IPv6
All other values are reserved	

- MEF Address: Address of the M2M Service Bootstrap Function.
  - This field shall be set to the address of the M2M Enrolment Function. When the MEF type is set to 0x00, the corresponding MEF Address shall be encoded to an octet string according to UTF-8 encoding rules as specified in IETF RFC 3629 [19].

Unused bytes shall be set to 'FF'.

D.2 oneM2M Service Module application for symmetric credentials on UICC (1M2MSM)

D.2.0 Introduction

This clause defines the oneM2M Service Module (1M2MSM), an application used for oneM2M Service Layer security functionalities and subscription provisioning based on symmetric keys. This application resides on the UICC, an IC card specified in ETSI TS 102 221 [24]. In particular, ETSI TS 102 221 [24] specifies the application independent properties of the UICC/terminal interface such as the physical characteristics and the logical structure. There may be several 1M2MSM ADFs on a single UICC, corresponding to independent oneM2M Service Subscriptions.

D.2.1 oneM2M Service Module application file structure

D.2.1.0 Introduction

This clause specifies the EFs for the oneM2M service Layer defining access conditions, data items and coding. A data item is a part of an EF which represents a complete logical entity.

D.2.1.1 Content of UICC files at the Master File (MF) level

Files at the UICC MF level are application independent as specified in ETSI TS 102 221 [24]. Only the EF_{DIR} and EF_{ICCID} files are mandatory on UICC for the purpose of 1M2MSM applications. In any case all files shall be as specified in ETSI TS 102 221 [24].

D.2.1.2 Content of files at the 1M2MSM ADF (Application DF) level

The EFs in the 1M2MSM ADF contain oneM2M subscription related information that is required for M2M field nodes operating in an oneM2M environment. This ADF shall be selected using its AID and information in EF_{DIR}. The AID for 1M2MSM applications shall be constructed as specified in ETSI TS 101 220 [27].

NOTE: The ETSI RID can be used for oneM2M pending assignment of a oneM2M dedicated RID in ISO/IEC 7816-5 [i.11].

The File IDs '6F1X' (for EFs), '5F1X' and '5F2X' (for DFs) with X ranging from '0' to 'F' are reserved under the 1M2MSM ADF for administrative use by the card issuer.

The DF_{1M2M} substructure used to isolate the provisioning of network access dependent M2M service related information in a Network Access Application ADF is not needed for access network independent provisioning of an M2M service subscription in a 1M2MSM ADF. Therefore, all the EFs specified in clause D.1.3 shall be present at the 1M2MSM ADF level. The file structure of the ADF_{1M2MSM} is illustrated in figure D.2.1.2-1.

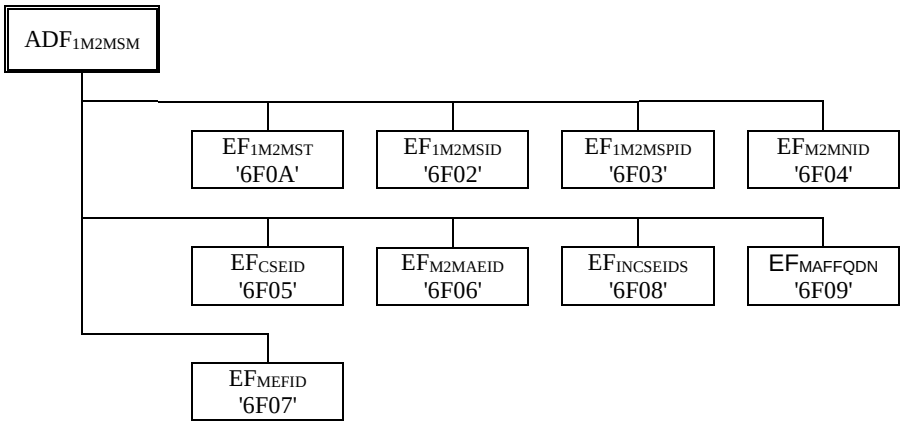


Figure D.2.1.2-1: File identifiers and directory structures of ADF_{1M2MSM}

D.2.2 oneM2M Subscription related procedures for M2M Service

D.2.2.0 Introduction

This clause specifies the procedures that shall be executed by M2M field nodes to interact with a oneM2M Service Subscription on UICC. They are applicable independently of the file structure supporting the oneM2M Service Subscription (1M2MSM ADF or DF_{1M2M} under a Network Access Application ADF), unless otherwise indicated.

D.2.2.1 Initialization - 1M2MSM Application selection

This procedure only applies to an M2M subscription supported in a 1M2MSM ADF.

If the M2M field node wants to engage in M2M operation, then after UICC activation (see ETSI TS 102 221 [24]), the M2M field node shall select a 1M2MSM application, if a 1M2MSM application is listed in the EF_{DIR} file, using the SELECT by DF name as defined in ETSI TS 102 221 [24].

After a successful oneM2M application selection, the selected oneM2M AID is stored on the UICC. This application is referred to as the last selected 1M2MSM application. The last selected 1M2MSM application shall be available on the UICC after a deactivation followed by an activation of the UICC.

If a oneM2M application is selected using partial DF name, the partial DF name supplied in the command shall uniquely identify a 1M2MSM application. Furthermore if a 1M2M application is selected using a partial DF name as specified in ETSI TS 102 221 [24] indicating in the SELECT command the last occurrence, the UICC shall select the oneM2M application stored as the last oneM2M application. If, in the SELECT command, the options first, next/previous are indicated, they have no meaning if an application has not been previously selected in the same session and shall return an appropriate error code.

D.2.2.2 1M2MSM session termination

This procedure only applies to a oneM2M subscription supported in a 1M2MSM ADF.

The oneM2M UICC session is terminated by the M2M field node as follows:

- The M2M field node shall indicate to the oneM2M UICC application that the termination procedure is starting, by sending a particular STATUS command.
- Finally, the M2M field node deletes all the M2M subscription related information elements from its memory.
- To actually terminate the session, the M2M field node shall then use one of the mechanisms described in ETSI TS 102 221 [24].

D.2.2.3 oneM2M Service discovery procedure

This procedure is used to discover the oneM2M related services offered by a oneM2M UICC.

The M2M field node shall perform the reading procedure with EF_{1M2MST}. If no oneM2M related service is indicated as available, the M2M field node shall assume that only the provisioning of mandatory parameters is available in this ADF.

D.2.2.4 oneM2M Service provisioning procedures

These procedures are used by an M2M field node in order to bootstrap an M2M service subscription provisioned on the UICC.

The M2M field node shall perform the reading procedure with $EF_{1M2MSID}$ and $EF_{1M2MSPID,}$ and EF_{CSEID}, EF_{M2MNID}, $EF_{INCSEID}$, $EF_{MAFFQDN}$ according to available services indicated in EF_{1M2MST}.

D.2.2.5 oneM2M Application Identifiers provisioning procedure

This procedure provisions a list of M2M Application Identifiers that may be enabled on the M2M node in relation with the oneM2M Service Subscription.

Condition: Service number 4 shall be available in the oneM2M Service Table.

Under this condition, the M2M field node shall perform the reading procedure with $EF_{M2MAEID}$.

D.2.2.6 oneM2M Secure provisioning related procedures

These procedures are used by the M2M field node to perform M2M Secure Provisioning with the assistance of the UICC, depending on available services in EF_{1M2MST} and the supported AUTHENTICATE commands contexts (e.g. GBA support by a Network Access Application) indicated for the hosting ADF.

Secure Provisioning: MEF address Provisioning:

Condition: Service number 5 shall be available in the oneM2M Service Table.

Under this condition, the M2M field node shall perform the reading procedure with EF_{MEFID}, if the related service is available.

GBA Secure Provisioning:

This procedure is dependent on the Authentication Framework supported by the UICC and indicated in the Service Table of the hosting ADF.

After identifying the supported authentication framework, the M2M field node shall check availability of Service number 7 in EF_{1M2MST}: If the service is available, the D/G M2M Node shall perform GBA-related procedures with AUTHENTICATE - GBA security context (Bootstrapping Mode and Derivation Mode) with the parameters for GBA secure provisioning.

D.2.2.7 oneM2M Security Association related procedures

GBA secure connection:

This procedure is dependent on the Authentication Framework supported by the UICC and indicated in the Service Table of the hosting ADF.

After identifying the supported authentication framework, the M2M field node shall check availability of Service number 12 in EF_{1M2MST}: If the service is available, the M2M field node shall perform a GBA-related procedures with AUTHENTICATE - GBA security context (Bootstrapping Mode and Derivation Mode) with the parameters for GBA Security Association.

Annex E (informative): Precisions for the UICC framework to support M2M Services

E.0 Introduction

The present annex provides further practical information related to the UICC framework for oneM2M described in annex D.

E.1 Suggested content of the EFs at pre-personalization

If EFs have an unassigned value, it may not be clear from the main text what this value should be. This annex suggests values in these cases.

Table E.1-1: Pre-personalized EF values

File Identification	Description	Value
'6F02'	1M2M Service Subscription Identifier	'8000FF…FF'
'6F03'	1M2M Service Provider Identifier	'8000FF…FF'
'6F04'	M2M Node Identifier	'8000FF…FF'
'6F05'	Local CSE Identifier	'8000FF...FF'
'6F06'	M2M Application Identifiers list	'00FF...FF' for each record
'6F07'	MEF Identifier	'00FF…FF' for each record
'6F08'	IN-CSE Identifiers list	'00FF...FF' for each record
'6F09'	MAF FQDN	'8000FF...FF'
'6F0A'	1M2M Service Table	Operator/Service Provider dependant

E.2 EF changes via Data Download or CAT applications

This clause defines if changing the content of an EF by the UICC OTA protocol or by a CAT Application is advisable. Updating of certain EFs "over the air" or "over the Internet" could result in unpredictable behaviour of the UE; these are marked "Caution" in table E.2-1. Certain EFs are marked "No"; under no circumstances should "over the air/over the internet" changes of these EFs be considered.

Table E.2-1: EF update behaviour

File identification	Description	Change advised
'6F02'	1M2M Service Subscription Identifier	No
'6F03'	1M2M Service Provider Identifier	No
'6F04'	M2M Node Identifier	Caution
'6F05'	Local CSE Identifier	Caution
'6F06'	M2M Application Identifiers list	Caution
'6F07'	MEF Identifier	Caution
'6F08'	IN-CSE Identifiers list	Caution
'6F09'	MAF FQDN	Caution
'6F0A'	1M2M Service Table	Caution

E.3 List of SFI values at the ADF_{M2MSM} or DF_{M2M} level

Table E.3-1: SFI values

File Identification	SFI	Description
'6F02'	'02'	M2M Service Subscription Identifier
'6F03'	'03'	M2M Service Provider Identifier
'6F04'	'04'	M2M Node Identifier
'6F05'	'05'	Local CSE Identifier
'6F06'	'06'	M2M Application Identifiers list
'6F0A'	'0A'	1M2M Service Table

All other SFI values are reserved for future use.

E.4 UICC related tags defined in annex J

Table E.4-1: UICC tags

Tag	Name of Data Element	Usage
'80'	MAF FQDN TLV data object	$EF_{MAFFQDN}$
'80'	M2M-Node-ID TLV Data Object	EF_{M2MNID}
'80'	Local CSE-ID TLV data object	EF_{CSEID}
'80'	M2M-SP-ID TLV data object	$EF_{1M2MSPID}$
'80'	M2M Subscription Identifier TLV data object	$EF_{1M2MSID}$
NOTE: The value 'FF' is an invalid tag value.		

Annex F (normative): Acquisition of Location Information for Location based Access Control

F.0 Introduction

When a request (resource access) is evaluated by a Hosting CSE and an accessControlLocationRegions parameter is defined in the privileges attribute of the <accessControlPolicy> resources, the Hosting CSE shall check whether the location of the Originator of a request is in the specified regions or not. Therefore, the Hosting CSE shall retain the location of the Originator, or acquire the location or deny the access. This annex indicates how to describe the location regions and obtain the location of the Originator.

F.1 Description of Region

F.1.1 Circular Description

The practical way of describing the region or area is the circular presentation and generally the circle is characterized by the co-ordinates of a centre point of the circle and a radius. Geographically, the centre point and radius is described as longitude and latitude, and meter respectively. For this description, the accessControlLocationRegions parameter shall be represented as a circle.

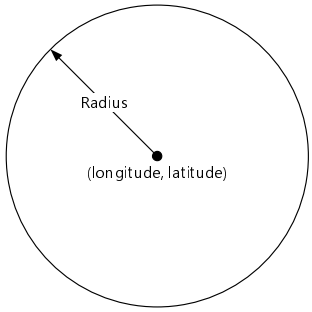


Figure F.1.1-1

F.1.2 Country Description

*Another simple way of describing the region or area is the country presentation. ISO-3166-1 alpha 2 codes [*i.10*] are two-letter country codes to represent countries and special regions of geographical interest. For example, KR is a code for Korea, Republic of.*

F.2 Acquisition of Location Information

F.2.0 Introduction

As mentioned above, when accessControlLocationRegions parameter is defined, the Hosting CSE shall check the location of the Originator for access control. This clause describes how the Hosting CSE checks or obtains the location. The procedures may vary based on the region description, circle and country.

F.2.1 Circular Description

If the circular description is used as the location context constraints, the Hosting CSE shall check whether it has the current location of Originator or not. If not, it shall obtain the location of Originator. ETSI TS 118 101 *[1] defines a resource type for acquisition of location of a Target Node, <locationPolicy>. Therefore, in order to obtain the location of Originator, the Hosting CSE shall create <locationPolicy> and set the relevant attributes as follows:*

- ***locationSource:*** *Reliability of the location information is crucial so the location shall be obtained from trusted network. If the location is obtained by the other sources, the location information can be easily masqueraded (i.e. GPS spoofing). Therefore, the locationSource attribute shall be set to 'network-based'.*
- ***locationTargetID:*** *The Target Node shall be the Originator that needs to authorize the sent requests. The locationTargetID attribute shall be set to identifier of the Originator.*

Note that the other attributes are determined by local policies of Hosting CSE as described in clause 9.6.9 of ETSI TS 118 101 [1]. In order to obtain the location from the network, the Hosting CSE shall transform the oneM2M specified location request into network specified request.

NOTE 1: ETSI TS 118 104 [4] describes how to convert the oneM2M-specified request to 'OMA RESTful NetAPI for Terminal Location' specified request, in annex F.

Since the region information (circular description) is defined by the accessControlLocationRegions parameter, the Hosting CSE can utilize the circular region information when it requests the location information from the network. OMA RESTful NetAPI for Terminal Location specification [i.9] specifies resource types as an area (region)-based location notification service, 'CircleNotificationSubscription'. If therefore the Hosting CSE subscribes to the notification service with circular region defined as acccessControlLocationRegions parameter, the Hosting CSE can always determine whether the Originator is in the regions or not. Figure F.2.1-1 demonstrates how to acquire the location of the Originator when the accessControlLocationRegions parameter is defined.

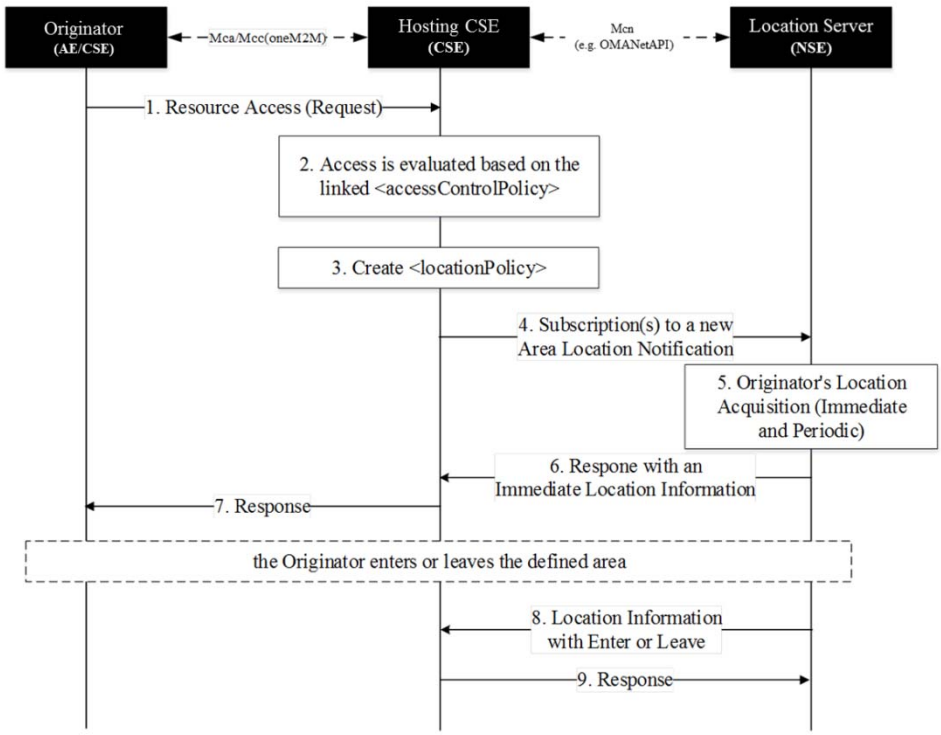


Figure F.2.1-1

1. *The Originator sends a request to access a resource.*

2. *The Hosting CSE shall evaluate the received request against the linked <accessControlPolicy> resource. If one of the rule tuples that is about the request originator contains the accessControlLocationRegions parameter (circular description) and the Hosting CSE does not store the location of the Originator, the Hosting CSE shall either continue to the next step or deny the access.*
   If the Hosting CSE has the location of the Originator, it is used for applying access control policy.

   The Hosting CSE may deny the access due to the fact that the Originator is not subscriber of the network or any other reasons (e.g. connection lost, server malfunction).

3. *The Hosting CSE shall create the <locationPolicy> resource and set relevant attributes as mentioned above.*

4. *The Hosting CSE shall subscribe to a new area location notification service toward Location Server in the Network. The area information shall be based on the area defined by the accessControlLocationRegions parameters. If multiple regions are defined, multiple subscriptions shall be set.*

5. *The Location Server immediately obtains the location of the Originator.*

NOTE 2: After the immediate location acquisition, the Location Server periodically obtains the location of the Originator to check whether the Originator is in the area or not. The frequency and duration can be defined by local policies.

6. *The Location Server responds with the immediate location information of the Originator toward Hosting CSE.*

7. *Based on the received location of the Originator and other access control policies, the request is evaluated and can be either granted or denied. The Hosting CSE responds regarding the request (step 1).*

8. *When the Originator crosses in (enter) or out of (leave) the area, the Location Server shall notify of the Hosting CSE of the location change. Thus, the Hosting CSE can keep track of the location of the Originator and easily evaluate the access against location context constraint.*

9. *The Hosting CSE responds to the notification.*

F.2.2 Country Description

Generally, the Originator's country-scale location can be determined by the Originator's IP address. If the Hosting CSE can distinguish the country using the Originator's IP address and it is also matched with the defined acccessControlLocationRegions parameter, the Hosting CSE may grant the request subject to evaluation of the full access control policies.

NOTE 1: How to transform the IP address into country is out of scope.

However, if Hosting CSE cannot distinguish the country using the Originator's IP address, the Hosting CSE shall obtain the location coordinate (i.e. longitude and latitude) of the Originator from the network and the Hosting CSE can distinguish the country using the location if available. The way of obtaining the location coordinate is defined in annex F of ETSI TS 118 104 [4].

NOTE 2: How to transform the location into country is out of scope.

Annex G (informative): Access Control Decision Request

An Access Control Decision Request as introduced in the Authorization Architecture in clause 6.2.2 is generated by a PEP according to an Originator's access request and extra information provided by the hosting CSE using the format specified by the PDP. The PEP can send the Access Control Decision Request to a PDP for an access control decision.

The PDP asks the PRP to retrieve all applicable access control policies according to the Access Control Decision Request, and then uses the Access Control Decision Request to evaluate the retrieved access control policies for an access control decision. An Access Control Decision Request from PEP to PDP can contain the following information:

- An Originator: It represents the ID of the Originator that sends an access request to the target resource.
- A Resource: It represents the URI of the target resource which the Originator wants to access.
- An Operation: It represents the operation which the Originator wants to perform on the target resource.
- An AccessTime: It represents the time of access.
- A LocationRegion: It represents the location of the Originator.
- An IPAddress: It represents the IP Address of the Originator.

The URI of the target resource is used to locate the target resource and then find the associated access control policies.

The ID of Originator is used to compare with the rule component subjects in order to check if a rule is applicable to the Access Control Decision Request.

The operation is used to compare with the rule component operations in order to check if the operation is permitted by the rule.

The AccessTime, LocationRegion and/or LocationRegion are used to check the rule component contexts in order to ensure some extra conditions are satisfied to using the rule for making an access control decision.

Annex H (informative): Implementation Guidance and index of solutions

The use of the present document involves a context-specific risk assessment process from which relevant security solutions are identified.

Clause 6 provides an overview of oneM2M security procedures. Clause 6.1.1 presents the interactions between layers, clause 6.1.2 introduces the sequence of events, and clause 6.2 provides further background especially for authorization (clause 6.2.2).

Clause 7 on Authorization and Access Control applies regardless of the type of credentials used. Clause 7.1 describes the general oneM2M Access Control Policy management framework, which can be further enhanced by supporting frameworks for Role-based Access Control (clause 7.4), Dynamic Authorization (clause 7.3). In addition, clause 11 leverages on the above to provide a Privacy Protection Architecture that facilitates the setting and management of user's privacy profiles.

The present annex provides a table to assist implementers in identifying which clauses of the present document are relevant for a given type of credential. Specific clauses that apply for supporting End-to-end security are listed in italic characters.

Table H-1: Index of clauses specifying procedures per credential types

Procedure/Solution	PSK	Certificates	TEF (GBA, MEF, MAF)
Remote security provisioning	8.3.2.1	8.3.2.2, *8.7*	8.3.2.3 (GBA)
	9.2.1.1		9.2.1.2, 9.2.2.3, 9.2.2.4 (GBA), *9.2.3*
Security Association Establishment	8.2.1, *8.4 (ESPrim), 8.5.2.3 (ESData Sign), 8.5.2.4 (ESData Sign+Encrypt)*		
	9.1.1.1, 9.1.2.1		9.1.1.2, 9.1.2.2 (MAF)
	8.1.1, 8.2.2.1, *8.5.2.2.2 (ESData Encrypt)*	8.1.2, 8.2.2.2, *8.5.2.2.4 (ESData Encrypt)*	8.1.3, 8.2.2.3 (MAF), *8.8, 8.5.2.2.3 (ESData Encrypt)*
Algorithm details	10.2.1, *10.3.6*		
	10.2.2, 10.3	10.1, 10.2.3, 10.3	

Annex I:
Void

Annex J (normative): List of Privacy Attributes

Table J-1

Tag ID	Tag Name	Value	Parameter	Tag description (short form)	full Tag description	Notes
1.0	Who	Null	Null	Name of party	The trading name of the device or service provider asking for access to the users smart devices/network/data	
		variable	Txt	company name	The name of the company that is requesting access to the user's smart devices and specifying their terms	
		variable	Country code	Location	The country where the device or service provider is located	
		variable	Txt	Company Registration number	Company Registration number which can be used as an aid to check the authenticity of the company. For example you could use the UK company registration number available from the UK Companies House. Other equivalent country registration authorities can be used as an aid to check the authenticity of the company asking to use the data and how much trust to place in it	
1.1	ID				Options for how applications and devices are uniquely identified	
			Txt	Model number	These are the mobile number(s) of the device(s) if included in the ASP's service	
			Txt	Version	These are the version number(s) of the device(s) if included in the ASP's service	
			oneM2M Defined format	Registered App ID	These are the Registered App ID of the apps if included in the ASP's service	
			Country codes	Country codes where approval has been granted if needed	Device or app accreditation may only be valid in certain countries	
2.0	What			Data Classification Type	What is the type of data that the device/service will access? With the higher the value the more sensitive the data is	
		Data not collected	Yes/No	No data collected	the device does not gather any data, this could be output device, such as a light switch, that only receives instructions	
		Non personal data	Yes/No	data not linked to a person	The data cannot be linked to a person, this could be applicable if the device was a door sensor that can only report then it was opened or closed	

Tag ID	Tag Name	Value	Parameter	Tag description (short form)	full Tag description	Notes
		Anonymized data	Yes/No	Data is collected about a person but anonymized	Data is collected about a person but anonymized to remove or summarize any data that would allow an individual to the identified/profiled	
		Personal data	Yes/No	Data that can be directly linked to an identify	The data gather can be linked to an identifier that is unique to an individual or small group (e.g. family members in the same home)	
		Sensitive personal data	Yes/No	Data that can be linked to identify, of a more sensitive nature.	The EU DPA defines certain types of sensitive person data. Additional types such as banking should also be considered to see if they fall within this area	
		Medical data	Yes/No	Data related to an individual's health/fitness, etc.	Data about illnesses, treatments or general wellbeing	
3.0	When	Null	null	When the data is collected	How often data is sent	
		Data not collected	Yes/No	No data collected	The device/service does not collect data, e.g. an end device such as a light	
		Event based	Yes/No	Triggered by an event	Device only gathers data when triggered, such as a door sensor triggering a camera	
		Variable Monthly	1 to 12	data Is sent monthly	The device/services only gathers the data as a monthly transfer. For example a smart fridge, sending a routine operational status report	
		Weekly	Yes/No	data is sent weekly	The device/service only gathers the data as a weekly transfer. For example a diagnostic status report from your fire detection system, including sensor test results, predicted remaining battery life	
		Daily	Yes/No	data is sent daily	The device/service only gathers the data on a daily transfer. For example a smart fridge sending the items that have run out as a Grocery list to the users preferred retailer so the retailer can short list them for inclusion in the users shopping basket	
		hourly	1 to 24	Data is sent every X hours	The device/service only gathers the data on an hourly transfer. For example a house alarm reporting it is armed and all sensors have reported they are active. So user alarm app/alarm monitoring service knows that system is still operational and someone has not been able to disable the alarm ability to send an alert	
		minutes	1 to 60	data is sent every X minutes	The device/service only gathers the data every 15th minutes. For example smart meters reporting back their usage figures	
		Real time-triggered	Yes/No	The data is sent continuously then triggered.	The data is sent in real time, when a specific event triggers it. E.g. The house alarm reports an internal door opening while alarm it set, this triggers the streaming of security cameras	
		Real time-full	Yes/No	The data is sent continuously at all times	The data is sent in real time for the duration of the device being active. For example CCTV data being sent to offsite storage	
3.1	Time period	Null		time period of data	When data is sent, does it cover a time period	

Tag ID	Tag Name	Value	Parameter	Tag description (short form)	full Tag description	Notes
		Data not collected	Yes/No	No data collected	The device/service does not collect data, e.g. an end device such as a light	
		summary/current status	Yes/No	The device send its current status	The Device sends the current status data, with no history e.g. the current status of a door sensor (open/closed) and not the log of when the door was opened and closed	
		Sample	Yes/No	The data covers a short period of time.	The data covers a sample of data from a short period of time, such a periodic sampling of heart rhythm being sampled several times a day	
		full history	Yes/No	The full data captured by the device is provided	The full data captured by the device is provided, either sent in real time (3.0) or history uploaded retrospectively	
3.2	Sample rate	Null		the time period between data sampling	How long in seconds between samples been taken	
		Data not collected	Yes/No	No data collected	The device/service does not collect data, e.g. an end device such as a light	
		Variable		value is seconds between data capture points	How long between readings that the device takes measured in seconds	
		Streamed data	Yes/No	Data is captured continuously	Data is captured continuously, such as a smart security camera able to stream the feed to the user	
4.0	Where - stored	Null		Where the data is stored	Were the data created by the device or used by the service is stored	
		Data not collected	Yes/No	No data collected	The device/service does not collect data, e.g. an end device such as a light	
		Local	Yes/No	The data is only stored locally	The data is stored within the network of smart devices (e.g. within the home)	
		variable		Country/bloc	The country were the data is stored, or if part of a wider framework (such as the EU)	
4.1	Where - collected			Where the data is collected from.	Where the data is collected from -note this may be redundant for consumer, but could be used for external feeds such as weather reports. May also be relevant to services so they can state the tries of devices they will pull data from, as they may not want access to all smart devices in the location	
		Data not collected	Yes/No	No data collected	The device/service does not collect data, e.g. an end device such as a light	
		Device	Yes/No	Data is collected just the specific device covered by T&C	The terms & conditions (also well as users privacy settings) are only be evaluated against the data collected by the specific device	
		Smart device network	Yes/No	Data is collect from all devices on the users network	The data is collected from all the devices* that form the users smart device network	

Tag ID	Tag Name	Value	Parameter	Tag description (short form)	full Tag description	Notes
		variable		External feed	Data comes from an external feed, and is combined with data gather. E.g. Weather forecasts combined with users building utilization patens to predict, then to turn heating up so the building is at the desired temperature when the user arrives	This would be descriptive and the user would have two options. Disable or substitute (e.g. they have their own compatible weather station, instead of getting a feed from the meteorological office
4.2	Where - Processed	Null		Where is the data processed	Where (physical location) the data is processed. This may be different from the storage location	
		Data not collected	Yes/No	No data collected	The device/service does not collect data, e.g. an end device such as a light	
		Local	Yes/No	The data is only processed locally	The data in only processed on the device, or with the user's network of smart devices	
		Variable		Country/bloc	The country were the data is stored, or if part of a wider framework (such as the EU)	
4.3	Where - Accessed	Null		Were the data is accessible from	Where the supplier/vendor/Policy Precedence restrictions allow the data stored to be accessed from	
		Data not collected	Yes/No	No data collected	The device/service does not collect data, e.g. an end device such as a light	
		Local	Yes/No	The data is only processed locally	The data in only processed on the device, or with the user's network of smart devices	
		Variable		Country/bloc	The country were the data is stored, or if part of a wider framework (such as the EU)	
5.0	Why	Null			The prime reason why personal data is being collected and to allow any change of use to be notified to the user	
		Yes/No	Yes/No	For Direct Delivery of the service	The ASP collects information for the direct delivery of the service	E.g. using location for paging from a base station that the user is currently registered at
		Yes/No	Yes/No	To improve ASP's and their partners products and services	The ASP collects information to improve ASP's and their partners products and services	
		Yes/No	Yes/No	To personalize services	The ASP collects information to personalize ASP's and their partners products and services	"our customers that selected this also selected these"
		Yes/No	Yes/No	A Policy Precedence requirement	The ASP collects information to meet a Policy Precedence requirement	E.g. Minimum age of intended user used to determine access to resources
6.0	Retention	Null		How long is data retained	How long the data (defined above) is kept in its current level of detail	
		Data not collected	Yes/No	No data collected	The device/service does not collect data, e.g. an end device such as a light	
		Zero retention	Yes/No	Zero data retention	After processing data, its immediately deleted	

Tag ID	Tag Name	Value	Parameter	Tag description (short form)	full Tag description	Notes
		Minutes	1 to 60	Data is kept for X minutes	Data is kept for 15 minutes before being deleted. E.g. the device only holds the last set of readings and collects new ones every 15 minutes	
		hour	1 to 24	Data is kept for X hour	Data is kept for X hours	
		Day	1 to 7	data is kept for X day		
		week	1 to 4	Data is kept for X week		
		Month	1 to 12	Data is kept for X month		
		Year	1 to 10	data is kept for X year		
		forever		The data will be kept for ever	The data will be stored without a defined retention/deletion policy	
6.1	retention - anonymized	null		how long anonymized data is kept	How long any anonymized or other derived data that is not directly linked to a unique identify is kept. E.g. stats on power usage by geo location	
		Zero retention	Yes/No	Zero data retention	After processing data, its immediately deleted	
		Minutes	1 to 60	Data is kept for X minutes	Data is kept for 15 minutes before being deleted. E.g. the device only holds the last set of readings and collects new ones every 15 minutes	
		hour	1 to 24	Data is kept for X hour	Data is kept for X hours	
		Day	1 to 7	data is kept for X day		
		week	1 to 4	Data is kept for X week		
		Month	1 to 12	Data is kept for X month		
		Year	1 to 10	data is kept for X year		
		forever		The data will be kept for ever	The data will be stored without a defined retention/deletion policy	
6.2	retention - summary	Null		how long summary data is kept	How long summary data is kept, e.g. how much total power was used per month based on meter readings taken every 15 minutes	
		Zero retention	Yes/No	Zero data retention	After processing data, its immediately deleted	
		Minutes	1 to 60	Data is kept for X minutes	Data is kept for 15 minutes before being deleted. E.g. the device only holds the last set of readings and collects new ones every 15 minutes	
		hour	1 to 24	Data is kept for X hour	Data is kept for X hours	
		Day	1 to 7	data is kept for X day		
		week	1 to 4	Data is kept for X week		
		Month	1 to 12	Data is kept for X month		
		Year	1 to 10	data is kept for X year		
		forever		The data will be kept for ever	The data will be stored without a defined retention/deletion policy	
7.0	Sharing -full	Null		Who the full data is shared with.	Who outside the company has access to the full data by type	
		Data not collected	Yes/No	Data is not shared outside the company	Data is not shared outside the company providing the device/service with not processing contracted out	

Tag ID	Tag Name	Value	Parameter	Tag description (short form)	full Tag description	Notes
		Group	No/scope and reason	Data is only shared with companies in the same group	Data is only shared within other companies in the same group	
		Infrastructure provider	Yes/No	Data is stored on 3rd party infrastructure	The data is stored on a separate company's servers e.g. the company providing the device/service uses a cloud provider for storage or processing	See note 1.
		Subcontractor	Yes/No	Data is shared with subcontractor(s)	Data is shared with one or more subcontractors who provide part of the service	See note 2.
		Other contracted parties ancillary functions	No/scope and reason	Data is shared with other parties under contract	Data is shared with other parties under contract that provide additional (non-core) functions to use/operation of device. Such as providing newsletters, marketing offers, etc. Local warranty repair places, etc.	
		Affiliate	No/scope and reason	Data is shared with other private entities	Data is shared with other parties who have no direct or indirect involvement in the device/service. E.g. device suppliers sharing data with channel partners so they can target campaigns	
		Public bodies	No/scope, reason, bodies	Data is shared with key public bodies	Data is shared then certain conditions with certain bodies. E.g. on triggering of an alarm, data for your security devices is shared with police so they can respond in a suitable fashion. Or if a medical alarm goes off the ambulance service/hospital, etc. are sent details so they can respond	
7.1	Sharing - anonymized	Null		Who the data is shared with.	Who outside the company has access to anonymized data by type of user	
		Data not collected	Yes/No	Data is not shared outside the company	Data is not shared outside the company providing the device/service with no processing contracted out	
		Group	No/scope and reason	Data is only shared with companies in the same group	Data is only shared within other companies in the same group	
		Infrastructure provider	Yes/No	Data is stored on 3rd party infrastructure	The data is stored on a separate company's servers. E.g. the company providing the device/service uses a cloud provider for storage or processing	
		Subcontractor	Yes/No	Data is shared with subcontractor(s)	Data is shared with one or more subcontractors who provide part of the service	
		Other contracted parties ancillary functions	No/scope and reason	Data is shared with other parties under contract	Data is shared with other parties under contract that provide additional (non-core) functions to use/operation of device. Such as providing newsletters, marketing offers, etc. Local warranty repair places, etc.	
		Affiliate	No/scope and reason	Data is shared with other private entities	Data is shared with other parties who have no direct or indirect involvement in the device/service. E.g. device suppliers sharing data with channel partners so they can target campaigns	

Tag ID	Tag Name	Value	Parameter	Tag description (short form)	full Tag description	Notes
		Public bodies	No/scope, reason, bodies	Data is shared with key public bodies	Data is shared then certain conditions with certain bodies. E.g. on triggering of an alarm, data for your security devices is shared with police so they can respond in a suitable fashion. Or if a medical alarm goes off the ambulance service/hospital, etc. are sent details so they can respond	
7.2	Sharing - summary	Null		Who the data is shared with.	Who outside the company has access to summary data by type of user	
		Data not collected	Yes/No	Data is not shared outside the company	Data is not shared outside the company providing the device/service with no processing contracted out	
		Group	No/scope and reason	Data is only shared with companies in the same group	Data is only shared within other companies in the same group	
		Infrastructure provider	Yes/No	Data is stored on 3rd party infrastructure	The data is stored on a separate company's servers. E.g. the company providing the device/service uses a cloud provider for storage or processing	
		Subcontractor	Yes/No	Data is shared with subcontractor(s)	Data is shared with one or more subcontractors who provide part of the service	
		Other contracted parties ancillary functions	No/scope and reason	Data is shared with other parties under contract	Data is shared with other parties under contract that provide additional (non-core) functions to use/operation of device. Such as providing newsletters, marketing offers, etc. Local warranty repair places, etc.	
		Affiliate	No/scope and reason	Data is shared with other private entities	Data is shared with other parties who have no direct or indirect involvement in the device/service. E.g. device suppliers sharing data with channel partners so they can target campaigns	
		Public bodies	No/scope, reason, bodies	Data is shared with key public bodies	Data is shared then certain conditions with certain bodies. E.g. local councils gathering average water usage by geo-location	
8.0	informing	Null				
		Y/N	Y/N	T&Cs sent to email address registered by the end user	The device vendor application providers send their tag values in this table to an email address registered by the end user	
		URL	ACME.com/English/device type/model/T&C	T&Cs by displayed URL	The device vendor application providers make their tag values in this table available at a URL	Could be automatically processed by the PPM portal
		URL	ACME.com/English/device type/model/T&C	T&Cs by URL stored in device	The device vendor application providers make their tag values in this table available at a URL stored in the device	Could be automatically processed by the PPM portal and or device
		Y/N	Y/N	T&Cs on local Screen (if present)	The device vendor application providers make their tag values in this table available on the device screen (if present)	
		Y/N	Y/N	On remote screen associated with user	The device vendor application providers make their tag values in this table available on remote screen associated with user	

Tag ID	Tag Name	Value	Parameter	Tag description (short form)	full Tag description	Notes
		Y/N	Y/N	By post	The device vendor application providers send their tag values in this table to an postal address registered by the end user	
		Y/N	Y/N	SMS (txt)	The device vendor application providers send their tag values in this table to an SMS number registered by the end user	
9.0	Obtaining consent	Null				
			Y/N	Consent by In app default	User has to accept default by a single click in the app	
			Y/N	Consent by End user signed document	Summary XML signed with end users private key e.g. digital signature	
			Y/N	Consent by oneM2M recommended method	A oneM2M recommended method {TBA}	If this is defined in the future
10.0	Protection	Null			Five levels to describe how well that the end user privacy and security is protected are defined. Level 1 is the lowest and Level 5 the highest. Each of these levels provides requirements expected for a claim at that level	These levels align with those already proposed by oneM2M WG4
		Protection level claimed	1	Protection level claimed = 1	Level 1: lowest level with minimal claims that the end user privacy and security is protected. This level is used when minimum risk is associated with a breach of end user security and privacy	
		Protection level claimed	2	Protection level claimed = 2	Level 2: provides some level of confidence that the end user privacy and security is protected. Entities prove, through a secure authentication protocol, that the entity has control of the sensitive data/credentials. Controls are in place to protect against attacks on stored sensitive data/credentials	
		Protection level claimed	3	Protection level claimed = 3	Level 3: provides high confidence that the end user privacy and security is protected. This level is needed when substantial risk is associated with breach of end user security and privacy. Multi-factor authentication is used. Any sensitive data or information exchanged in authentication protocols is cryptographically protected in transit and at rest	
		Protection level claimed	4	Protection level claimed = 4	Level 4: provides very high confidence that the end user privacy and security is protected. This level is used when high risk is associated with a breach of end user security and privacy. This level provides the highest level of end user security and privacy. In addition to Level 3 this level requires the usage of tamper resistant hardware devices for the storage of all sensitive data such as cryptographic keys	
		Protection level claimed	5	Protection level claimed = 5	As level 4 but evidence of external accreditation/assurance available	May depend on work of oneM2M on device certification

Tag ID	Tag Name	Value	Parameter	Tag description (short form)	full Tag description	Notes
11.0	Age	Null			When relevant to privacy settings options for how the end users age can be determined e.g. DOB age range, etc.	
		DD/MM/YYY		Date Of Birth	Claimed Date Of Birth of the end user to determine access to resources	
		Numeric value or range of values		Minimum age of intended user	Minimum age of intended user used to determine access to resources	
		DD/MM/YYY		Maximum age of device (shelf life)	Maximum age of device (shelf life) used to determine access to resources	For devices with embedded batteries that have a shelf life or chemicals in medical devices, etc.
NOTE:1: The answers for where reflect the 3rd party as well as the company offering the service/device. NOTE 2: The answers about location, etc. reflect all subcontractors used.						

Annex K (informative): Terms and Conditions Mark-up Language implementation rules

Typical implementation rules are shown below and are repeated for each row.

Note on conventions: {} identifies the answers to earlier if statement, [] identifies the Filter Frame and () contain comments. The logic has been shown with indents to better show the nesting of the statements. The logic works by checking the same rows on the Filter Frames being checked.

To generate the summary value for each row the follow logic is used.

```
If [Current Filter Frame] Value, is not equal to NA (Not applicable= No preference or limit set)?
    {Yes} Is [Current Filter Frame] value equal to [Previous Filter Frame] summary value (compound
value) of?
        {Yes} [Current Filter Frame] Summary value equals Value set.
        {No} is [current Filter Frame] Value set ="Yes"?
            {Yes} [Current Filter Frame] Summary value set as [Previous Filter Frame] Summary value
            {No} [Current Filter Frame] Summary value set as [Current Filter Frame] Summary value.
    {No} [Current Filter Frame] summary value set as [Previous Filter Frame] Summary value
To generate the T&C acceptable symbol the following logic is used.
If [Current Filter Frame] Value equals "Yes"?
    {Yes] [Current Filter Frame] T&C Acceptable set as "☺" (the smiley is used so the result
displayed to the user is language agnostic as well as only requiring a small amount of screen
space).
    {No} [Current Filter Frame] Value equal [Previous Filter Frame]?
        {Yes} [Current Filter Frame] T&C acceptable set as "☺"
        {No} [Current Filter Frame] T&C Acceptable set as "☹"
```

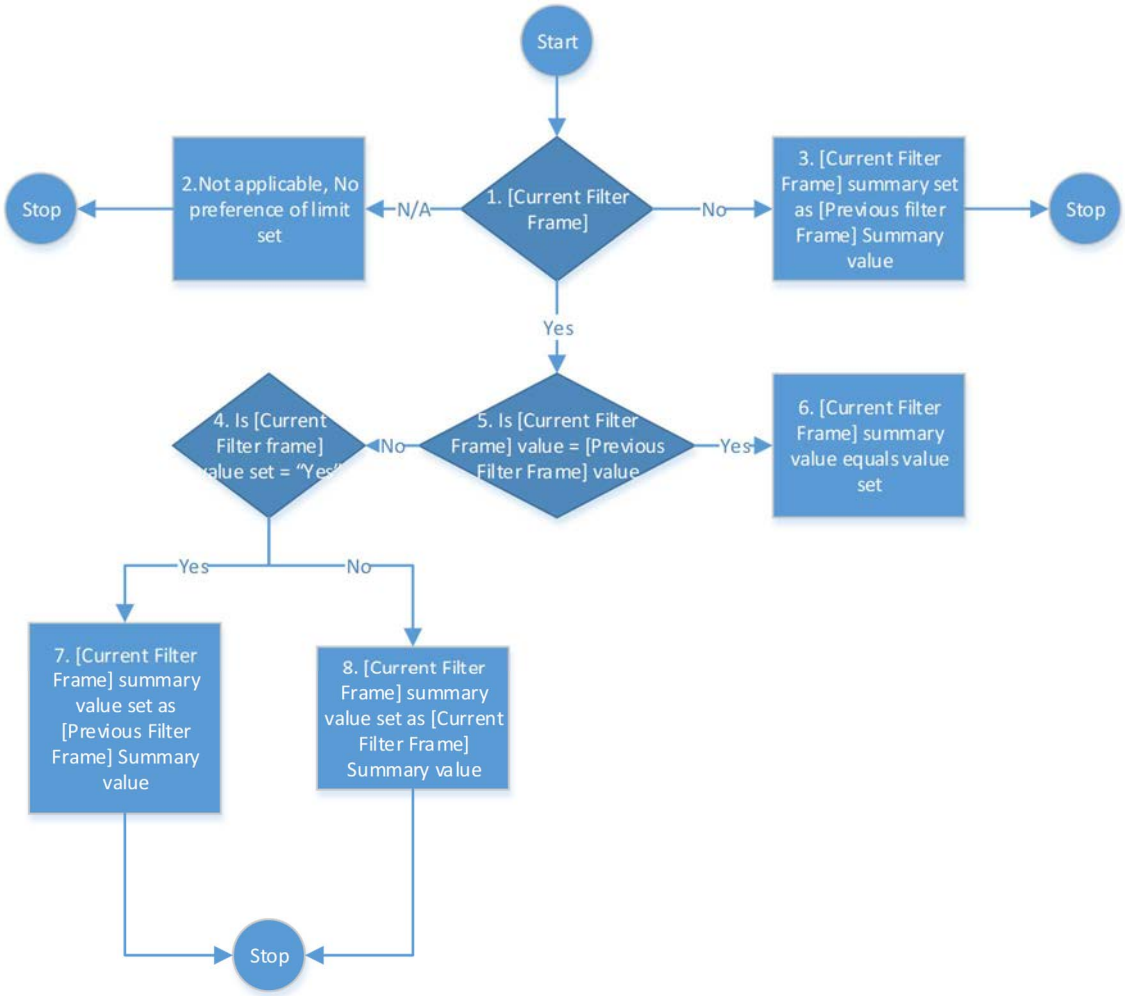

Start
1. [Current Filter Frame]
N/A
2.Not applicable, No preference of limit set
Stop
No
3. [Current Filter Frame] summary set as [Previous filter Frame] Summary value
Stop
Yes
5. Is [Current Filter Frame] value = [Previous Filter Frame] value
Yes
6. [Current Filter Frame] summary value equals value set
No
4. Is [Current Filter frame] value set = "Yes"
Yes
No
7. [Current Filter Frame] summary value set as [Previous Filter Frame] Summary value
8. [Current Filter Frame] summary value set as [Current Filter Frame] Summary value
Stop

Figure K-1

Annex L (normative): Tamper-resistant Secure Element framework supporting asymmetric cryptography services

L.0 Introduction

L.0.1 Overview

Secure elements may be integrated in PKI systems to provide secure identification and authentication of devices, tamper-resistant storage and execution areas for sensitive data (especially secure storage of private keys which may be generated on board in the SE and always used within it) managed by defined stakeholders, and digital signature services with management of digital certificates. Secure Element supporting asymmetric cryptographic services are termed Asymmetric Secure Element (ASE) in the rest of the present annex, which specifies features that should be exposed by the ASE to its hosting device to enable interoperable application deployments:

- Providing keys (which may be randomly generated data) to the hosting device for encrypting, integrity protecting and authenticating data sent by the hosting device to receiver of the data.
- Negotiation of keys for protecting the communication between hosting device and ASE.
- Calculating signatures for data to provide non repudiation.
- Generation of random numbers for the TLS command ClientHello.
- Key negotiation of the TLS pre-master secrets.
- Signature generation and verification for the TLS authentication.
- Providing generic cryptographic services to Application Entities.

The ASE may be a UICC [24] or eUICC [70], in which case the framework proposed in the present annex may coexist with some features specified in annex D, e.g. by being implemented as a GlobalPlatform applet loaded on a UICC. Other types of ASE may exist (e.g. embedded Secure Element according to GlobalPlatform).

The ASE capabilities specified in the present annex may be implemented as a secure element applet as per GlobalPlatform Card Specifications [63], which first needs to be selected in order for the ASE to exhibit the specified behaviour. This implementation provides the possibility to install and provision the asymmetric cryptographic capabilities on secure elements, even after deployment on the field, in a standard manner. It also enables to leverage on the Security Domains structure (SD) of the GlobalPlatform Card specification [63], allowing multiple stakeholders to independently operate and securely manage their own secure environments on a single Secure Element.

L.0.2 Naming Conventions

To easily identify whether a key is public or private, whether it exists in the ASE or the hosting device or is a CA key, and also the usage of a key, the following notation is used in this annex:

KeyType.KeyOwner.KeyUsage

To easily identify whether a certificate can be verified in the ASE or not, whether it exists in the ASE or the hosting device or belongs to a CA or root CA, and also its usage, the following notation is used in this annex:

CertType.CertOwner.CertUsage

The possible values are shown in table L.0.2-1.

Table L.0.2-1: Naming convention

Parameter	Value	Meaning
Key or certificate Type	PuK	Public Key
	PrK	Private key
Owner	ICC	ASE
	IFD	Hosting device (i.e. interface with M2M application)
	CA	Certification Authority
	CA_{ICC}	Certification authority that generated the certificate for the ICC public key
	RCA	Root Certification Authority
Usage	AUT	Authentication key
	DS	Digital signature key
	KA	Key Agreement
	CS-AUT	Certificate Signature Authentication

L.1 Physical interface and transport protocol

The intention of the present annex is to specify a set of generic security services that shall be supported in oneM2M ASE and should be exposed to oneM2M applications through the Secure Environment Abstraction Layer of ETSI TS 118 116 [67].

The ASE services are described at a high level in order to support implementations that comply with specific regulations, e.g. regional standards such as BS EN 419 212-1 [64] in the European Union or FIPS 201-2 [69] in the USA, or vertical such as BSI TR 03109 [i.32] in the German energy sector. The ASE security services described in this annex are commonly supported in secure elements used for certificate-based security deployments, such as governmental or corporate identification cards supporting digital signature as per BS EN 419 212-1 [64] or FIPS 201-2 [69].

The functionalities described in the present annex imply the presence of a random number generation capability in the ASE. This functionality may be made available to the hosting device. They also imply that the ASE supports asymmetric cryptography based on the RSA or ECC algorithms, and the AES symmetric algorithm.

The Secure Element may interface with the hosting M2M device through various physical communication means. The difference between the multiple communication links (wired or contactless) that may be used does not otherwise impact the way applications would interact with the Secure Element.

L.2 Lifecycle phases

The ASE lifecycle comprises the following phases:

- Personalization, where the ASE maintains the state initialized upon creation to enable its initial provisioning. This phase is supposed to take place in a trusted facility under control of the stakeholder responsible for the ASE (e.g. ASE issuer facility, device assembly line or Point of sale). It ends when the ASE receives a trigger to transition into its operational state.

- Operational phase, where the ASE maintains a state suitable for secure operation in the field, into which a transition is triggered upon completion of the personalization phase.

A secure channel shall first be established to secure data exchange with a host, as described in clause L.3. Depending on the operating environment, the secure channel may only ensure mutual authentication between both entities, or add MIC protection, or add both MIC and confidentiality protections.

Operation of the ASE (or ASE applet) during its personalization phase can be subject to specific constraints and can include special commands that are not available in the operational state. For example, the GlobalPlatform Card Specification [63] specifies low level personalization commands and procedures that may be implemented by ASE supporting ISO/IEC 7816-4 APDUs [26] in deployments requiring interoperability in the personalization state.

At the end of initial provisioning/personalization, the ASE (or ASE applet) enters an operational state, in which the functions specified in clause L.4 shall be available.

During operation, the secure element or specific information within it (e.g. keys) may move to a "blocked" state designed as a protection mechanism once it encounters any integrity problem or e.g. if a maximum allowed number of authentication attempts has been reached.

L.3 Device Application / ASE Authentication and Secure Channel Establishment

To prevent execution of commands and access to information by unauthorized entities, communication between the hosting device application and ASE shall be protected through the establishment of a secure channel (e.g. based on access control mechanism or mutual authentication of the communicating entities) both in the personalization and the operational state. This enables the protection of the information exchanged over the Mcs and Mca reference points. This mechanism ensures that:

- On one side, any entity (such as a clerk) which wants to access the protected data on the ASE, shall authenticate themselves to the ASE. Behind the entity are the system and the hosting device (called IFD). The ASE checks that the entity who is requiring access to the data is allowed to do so.
- On the other side, the ASE authenticates itself to the clerk's systems via the IFD, to ensure that it is genuine.

After mutual authentication between an entity and the ASE, the ASE grants the specific access rights related to the entity.

The secure channel authentication required for the ASE and external entity to exchange sensitive information may be based on either symmetric or asymmetric credentials:

- Asymmetric key mutual authentication based on the ASE and IFD verifying the existence of a certified key pair in the other entity. This process can be based on either RSA device authentication, or ECC device authentication. Where needed, common symmetric session keys can then be derived using the Diffie–Hellman key exchange mechanism to ensure integrity and/or confidentiality of the information exchange.
- Symmetric key mutual authentication based on the ASE and IFD verifying the existence of two AES symmetric secret keys, K_{ENC} and K_{MIC}, in the other entity. A successful symmetric mutual authentication opens the secure channel.

Establishment of a secure channel, i.e. a secure messaging session, requires a successful mutual authentication between the ASE and hosting device.

The following scenarios shall terminate a secure channel:

- Power off or reset of the ASE.
- Reselection of the ASE applet.
- A command with an incorrect MIC is received by the ASE.
- A command with an incorrect encryption is received by the ASE.

The present annex does not mandate any specific secure channel mechanism to allow alignment with contextual requirements. Example of relevant secure channel mechanisms include the following:

- Secure Channel Protocols (SCP) specified in the GlobalPlatform Card Specification [63], such as SCP 11 or SCP 03.
- Secure channel mechanisms specified in the GSMA eUICC technical specifications SGP.02 and SGP.22 [70].
- Secure Channel mechanisms specified in BS EN 419 212-1 [64] or FIPS 201-2 [69].

L.4 ASE Supported Functions

L.4.1 ASE Verifiable Certificates

These are certificates stored in the ASE and used in asymmetric key mutual authentication. The ASE Verifiable Certificate is issued and signed by a trusted certificate authority (CA) and stored in the hosting device to show that it (and so the entity behind it) can be trusted. This certificate is referred as C_CV.IFD.AUT. The ASE can check that the ASE Verifiable Certificate in the hosting device can be trusted by using the CA's public key.

Similarly, the ASE may contain a certificate issued and signed by the CA, called the C.ICC.AUT. The hosting device can check that this certificate was genuinely issued and signed by the CA by using the CA's public key.

In BS EN 419 212-1 [64], ASE Verifiable Certificates used in RSA-based device authentication are non self-descriptive (i.e. the tags and lengths of the signature elements are not included in the format), while ASE Verifiable Certificates used in Elliptic Curve Device Authentication are self-descriptive. Such ASE Verifiable Certificates include a Certificate Holder Authorization (CHA) that may be used as a security condition to access relevant sensitive data.

L.4.2 ASE Secure Storage

L.4.2.1 Overview

An ASE shall support a way to store information in its protected non-volatile memory. For example an ISO 7816 file system, or GlobalPlatform functionalities to store data inside a Secure Element without a file system.

This can be used for information meant to be exchanged with external entities: This includes permanent storage of stakeholder information, storage of service credentials, and storage of data for service processing. This can be updated dynamically during operation provided that access control conditions are satisfied.

Data objects are meant to store information used during internal processes such as secret keys. The structures for Data objects may need to be reserved during the personalization phase but their content can be updatable, if desirable, during the operational phase.

L.4.2.2 PIN

PINs may be used to identify a user and to protect data. See clause L.4.6 for further details.

L.4.2.3 Symmetric secret keys

Symmetric secret keys are 16-byte, 24-byte or 32-byte AES keys used for symmetric key mutual authentication. Two secret keys, KENC and KMIC, are shared by the secure element and its host, and can be diversified, for example by using the secure element serial number. Mutual authentication consists of each entity proving that it possesses the two keys to the other entity. A symmetric key can optionally be protected by a ratification counter. There may be multiple key pairs (KENC, KMIC) in an ASE. They shall be created together and initialized during the personalization phase.

L.4.2.4 Public keys

RSA and ECC public keys are associated with private keys in a key pair sharing a common one byte identifier, KID. These could be used for mutual authentication or to verify a signature or certificate. RSA Public Keys can also be used to encrypt sensitive data, while ECC Public Keys can be used to derive a symmetric shared key (ZZ) to be used to encrypt data.

The typical process to create a key pair in an ASE requires reservation of space for an Asymmetric Key Header during the personalization phase. This initializes a key container with at least a public portion and optionally a private portion.

The following public keys are generally stored in the ASE:

- CA public keys used in asymmetric key mutual authentication

- RSA and ECC public keys used by the application

More than one CA may store its public key PuK.CA.AUT on an ASE.

RSA public keys always contain a modulus, N, and a public exponent, e.g. the keys may be automatically updated by ASE internal process, or the keys may be generated outside the secure element.

L.4.2.5 Private keys

Private keys are used for public key cryptographic operations of M2M applications, such as generation of digital signatures, sensitive data decryption, and asymmetric scheme mutual authentication.

Private keys are always stored in the ASE to be adequately protected. They may be initialized either during the personalization phase or during the operational phase.

L.4.2.6 Diffie-Hellman Key Exchange parameters

The Diffie–Hellman key exchange parameters used in asymmetric key mutual authentication may also be stored in the ASE.

L.4.2.7 Arbitrary Application Data

This provides a service to create, store, update and delete application data in the SE.

L.4.2.8 ProfileData

This provides a service to store and protect profile data. A profile is the representation of parameters and data for its application, keys, and load files.

L.4.3 On-Board Key Generation (OBKG)

The On-Board Key Generation functionality enables creation of a public / private key pair within an ASE, so that the private key never leaves the ASE which protects it during storage and usage (e.g. to sign a certificate).

OBKG is initiated when a command is sent to the ASE to initialize or update the value of a key pair when the ASE is in Operational state. This command only generates new values for private key and public key and returns the public key value in its response.

On-Board Key Generation has several advantages:

- The ASE performs the computation of the key values. The key value is not precomputed or imposed by an external entity.
- As the key update takes place within the ASE, the secure element handles the security of the operation instead of the hosting application.
- The command may need to satisfy access conditions to the private key data object in order to update the value of the private key data object.
- The new private key value never leaves the secure element.
- The life span of the key pair can be easily managed within the application, e.g. by regular renewals in order to adapt to the specific risks to which the key pair may be exposed.

L.4.4 Digital Signature

L.4.4.1 Overview

The ASE may be used to generate Digital Signatures, by which a message is authenticated by the receiver to ensure that it is sent by the intended sender and that the message was not altered since it was sent. The signatures are generated using the Digital Signature keys stored in the ASE.

L.4.4.2 Digital Signature Generation

The digital signature generation process is the computation of the message signature using the digital signature private key on a pre-computed message hash digest. As the signature is generated using the sender's private key which is securely stored in the ASE, the message can only be sent by authorized sender and not by anybody else.

The digital signature creation process is as follows:

1) **Message Hashing.** The sender (Host Application) computes the hash of the original message using a hash algorithm. The host application calls a command to perform the hashing.

2) **Formatting Hash to Digital Signature Input (DSI).** The ASE pads the hash to the length and format indicated by the hashing command.

3) **Signature Creation.** The hash is ciphered with the sender's private key. The result is known as the signature.

4) **Digitally Signed Message Sending**. The signature is appended to the original message and sent.

L.4.4.3 Message Hashing

The generation of the hash may be performed in three ways:

1) performed entirely by the ASE using a dedicated command;

2) performed externally;

3) partially performed by the ASE and partially performed externally (in this case, the data is split).

For RSA Signatures, the ASE may use any of the following secure hash algorithms:

- SHA-256
- SHA-384
- SHA-512

For ECC signatures, the ASE may also use any of these SHA algorithms.

L.4.4.4 Formatting Hash to Digital Signature Input (DSI)

The generated hash is typically shorter than the required length of the Digital Signature Input and needs to be padded accordingly. The DSI needs to conform to a particular format, so the hash cannot be simply padded by adding a padding character. For this reason, the ASE is able to perform the necessary padding.

L.4.4.5 Signature Creation

The ASE uses the DSI to compute the digital signature upon instruction from its host.

The following algorithms are supported:

- ALG_ECDSA_SHA_256: Signature algorithm ALG_ECDSA_SHA_256 generates a 32-byte SHA-256 digest and signs/verifies the digest using ECDSA with the curve defined in the ECKey parameters - such as the P-256 curve specified in the Digital Signature Standards specification [71], FRP256V1 or brainpoolP256r1 curves all recommended in [74].

- ALG_ECDSA_SHA_384: Signature algorithm ALG_ECDSA_SHA_384 generates a 48-byte SHA-384 digest and signs/verifies the digest using ECDSA with the curve defined in the ECKey parameters - such as the P-384 curve specified in the Digital Signature Standards specification [71].
- ALG_ECDSA_SHA_512: Signature algorithm ALG_ECDSA_SHA_512 generates a 64-byte SHA-512 digest and signs/verifies the digest using ECDSA with the curve defined in the ECKey parameters - such as the P-521 curve specified in the Digital Signature Standards specification [71].

L.4.4.6 Integrity of the Data to be Signed

The ASE may check integrity of the data to be signed, as required by some signature certification schemes.

L.4.4.7 Digital Signature Verification

The digital signature verification typically involves decrypting the signature using the sender's public key and hashing the original message using the hashing algorithm. If the hashes are equal, the signature is valid.

As the signature is created using the sender's private key, it can only be verified by the sender's public key. By verifying the signature, the recipient has proof that the sender's private key was used to encrypt the message hash and that the message has not been altered.

Since this does not require a high level of security, this process is typically performed externally and the ASE is not involved in this operation.

The principle of digital signature verification is shown for informational purposes only:

1) The receiver uses the sender's public key to decrypt the signature and retrieve the message hash.
2) The receiver hashes the original message and compares it with the result obtained in step 1. If the two hashes match, then the sender is authentic.

L.4.5 Encryption and Decryption

L.4.5.1 Overview

Public key pairs may be used for encryption and decryption of sensitive data, typically symmetric session keys.

In the case of RSA, the public key of the receiver's RSA key pair is used to encrypt messages and the private key of the key pair stored in the ASE is used to decrypt the message. The external entity uses the ASE's public key to encrypt the message, which is not a sensitive operation, while the ASE uses the corresponding private key to decrypt the message internally using the PSO- Decipher (RSA use) decryption function. This process ensures that only the intended recipients can decrypt and read the message. Upon successful completion of the command, the ASE returns the deciphered message in the response.

In the case of ECC, the public key of the receiver (the ASE) is used to derive a shared key ZZ, which is used to encrypt and decrypt data. The key is generated by the ASE.

For security reason, it is strongly recommended to never use the same private key for deciphering and signing.

The following clauses provide an example of a message encryption and decryption process wherein the encrypted data is a one-time session key that has been used to encrypt another message.

L.4.5.2 RSA Message Encryption and Decryption

The message encryption process is performed by the message sender (external entity). The process includes the following steps:

1) Message Encryption. The message sender encrypts the document with a one-time session key. Typically, this is an AES session key.

2) Symmetric Key Encryption. The message sender encrypts the symmetric session key with the host application RSA public key with a specified padding, e.g. PKCS #1.

3) Message Sending. The message sender sends the encrypted session key and the encrypted message to the host application.

The message decryption occurs in the host application. The process includes the following steps:

1) Symmetric Key Decryption. Upon receiving the message, the host application instructs the ASE to decrypt the symmetric key. The ASE returns the decrypted symmetric key in the response.

2) Message Decryption. The host application decrypts the message using the symmetric key retrieved in step 1. This step is performed by the host application.

For security reasons, it is strongly recommended not to use the same private key for decryption and signing.

The messages to be decrypted may be protected by e.g. RSASSA PKCS#1 v1.5 algorithm or RSAES OAEP algorithms.

L.4.5.3 ECC Message Encryption and Decryption

Encrypting a Message (ECC):

The steps are as follows:

1) The sender derives a shared key, ZZ, from the ASE certified public key (yb) and the hosting device ephemeral private key (ra). This process involves generation of a random challenge.

2) The sender encrypts a message using ZZ.

Decrypting a Message (ECC):

1) The sender sends both the encrypted message and his/her public key (ya) to the ASE acting as the receiver.

2) The receiver uses ZZ to decrypt the message.

L.4.5.4 AES Message Encryption and Decryption

The following methods may be supported according to ETSI TS 118 116 [67]:

- ALG_AEAD_AES_128_GCM: The AEAD_AES_128_GCM authenticated encryption algorithm works as specified in IETF RFC 5116 [72], using AES-128 as the block cipher, by providing the key, nonce, and plaintext, and associated data to that mode of operation.

- ALG_AEAD_AES_256_GCM: This algorithm is identical to AEAD_AES_128_GCM, but with the following differences: K_LEN is 32 octets, instead of 16 octets, and AES-256 GCM is used instead of AES-128 GCM.

- ALG_AEAD_AES_128_CCM: The AEAD_AES_128_CCM authenticated encryption algorithm works as specified in IETF RFC 5116 [72], using AES-128 as the block cipher, by providing the key, nonce, associated data, and plaintext to that mode of operation.

- ALG_AEAD_AES_256_CCM: This algorithm is identical to AEAD_AES_128_CCM, but with the following differences: K_LEN is 32 octets, instead of 16, and AES-256 CCM is used instead of AES-128 CCM.

- ALG_AEAD_AES_128_CCM_8: The AEAD_AES_128_CCM_8 authenticated encryption algorithm is identical to the AEAD_AES_128_CCM algorithm (see Section 5.3 of IETF RFC 5116 [72]), except that it uses 8 octets for authentication, instead of the full 16 octets used by AEAD_AES_128_CCM (see Section 6.1 of IETF RFC 6655 [31]).

- ALG_AEAD_AES_256_CCM_8: The AEAD_AES_256_CCM_8 authenticated encryption algorithm is identical to the AEAD_AES_256_CCM algorithm (see Section 5.4 of IETF RFC 5116 [72]), except that it uses 8 octets for authentication, instead of the full 16 octets used by AEAD_AES_256_CCM (see Section 6.2 of IETF RFC 6655 [31]).

- ALG_AES_BLOCK_128_CBC_NOPAD: Cipher algorithm ALG_AES_BLOCK_128_CBC_NOPAD provides a cipher using AES with block size 128 in CBC mode and does not pad input data.
- ALG_AES_CBC_ISO9797_M1: Cipher algorithm ALG_AES_CBC_ISO9797_M1 provides a cipher using AES with block size 128 in CBC mode, and pads input data according to the ISO 9797 [73] method 1 scheme.
- ALG_AES_CBC_ISO9797_M2: Cipher algorithm ALG_AES_CBC_ISO9797_M2 provides a cipher using AES with block size 128 in CBC mode, and pads input data according to the ISO 9797 [73] method 2 (ISO 7816-4 [26], EMV'96) scheme.
- ALG_AES_CBC_PKCS5: Cipher algorithm ALG_AES_CBC_PKCS5 provides a cipher using AES with block size 128 in CBC mode, and pads input data according to the PKCS#5 scheme.

L.4.6 User Authentication through PIN

PINs are used to identify the owner of an ASE and to protect its data.

A data object in the ASE may be protected by a PIN. In this case, access to the object shall only be allowed upon successful verification of the PIN.

An ASE may also support an Activation PIN verification mechanism to prevent unauthorized use of the ASE before verification that the ASE is provided to the authorized owner.

The Activation PIN needs to be presented once only during the Operational Phase.

The ASE may also support a "Force PIN Change Before Signature" mechanism.

If the feature is activated after personalization and if the Digital Signature key is protected by a PIN, the PIN shall be changed at least once after personalization to make the signature functionality available.

L.4.7 TLS-Handshake

The ASE may provide services for the establishment of TLS channels (Handshake), including:

- Generation of random numbers for the TLS command ClientHello
- Key negotiation of the TLS pre-master secrets
- Signature generation and verification for the TLS authentication
- Securing the data sent via the negotiated TLS channel

The applicable cipher suites are listed in clause 10.2.

L.4.8 getSEFunctions

This service provides a list of available sensitive functions provided by the secure element.

L.4.9 Random numbers

This service provides random numbers to the hosting device.

L.4.10 Calculating MICs

This service calculates MICs. The following algorithm may be supported:

- ALG_AES_CMAC_128: Signature algorithm ALG_AES_CMAC_128 generates a 16-byte Cipher-based MAC (CMAC) using AES with blocksize 128 in CBC mode with ISO 9797 [73] M2 padding scheme.

- ALG_AES_MAC_128_NOPAD: Signature algorithm ALG_AES_MAC_128_NOPAD generates a 16-byte MAC using AES with blocksize 128 in CBC mode and does not pad input data.
- ALG_HMAC_SHA_256: HMAC message authentication algorithm ALG_HMAC_SHA_256. This algorithm generates an HMAC following the steps found in IETF RFC 2104 [33] using SHA-256 as the hashing algorithm.
- ALG_HMAC_SHA_384: HMAC message authentication algorithm ALG_HMAC_SHA_384. This algorithm generates an HMAC following the steps found in IETF RFC 2104 [33] using SHA-384 as the hashing algorithm.
- ALG_HMAC_SHA_512: HMAC message authentication algorithm ALG_HMAC_SHA_512. This algorithm generates an HMAC following the steps found in IETF RFC 2104 [33] using SHA-512 as the hashing algorithm.

L.4.11 Device Authentication

This service provides authentication of the hosting device, verifying the authenticity of remote entities and negotiating session keys for protecting the communication between the mutual authenticated entities.

Annex M (informative): Example SCEP implementation

M.1 Introduction

This annex provides a description of an implementation of the Simple Certificate Enrolment Protocol (SCEP).

M.2 Certificate Provisioning procedures using SCEP

Figure M.1 shows a high level outline of the SCEP procedures. The figure identifies the following building blocks of a certificate automation service using SCEP specified in IETF RFC 8894 [66].

- Profile Provisioning is the primary and authoritative actor in any automation system. Provisioning informs the *device's automation client*, and the PKI service – the credential issuer, though the establishment of pre-authorized device credentials, that a number of unique devices will be calling home to request dedicated unique client certificate(s).

- The provisioning capability informs both the remote device and the PKI service over an authenticated and confidential channel of their unique Provisioning Profiles. The Provisioning Profiles can be revised at any time, allowing existing credentials to be forced changed if necessary. Typical provisioning protocols include BBF TR-069, OMA-DM, etc., see [i.29] and [i.30].

- The *device automation client*, or certificate application intelligence provides a state machine that uses the provisioning data, a.k.a Provisioning Profiles, to generate keys and request and replace certificates at pre-determined periods in time by making requests of a native SCEP client. Typically the intelligence is time driven, ensuring timely renewal of existing keys and certificates; however it can also be event driven by the receipt of revised Provisioning Profiles from the provisioning system. The SCEP client is a native application installed on systems, servers or devices, it communicates with a SCEP responders using a protocol defined in IETF RFC 8894 [66]. The particular SCEP responder(s) are identified within the various Provisioning Profiles.

- The figure identifies a number of example SCEP message request response messages – these are documented within the IETF RFC 8894 [66]. The SCEP responder on receipt of a chain certificate request, responds by supplying the requested certificate. On receipt of a client certificate request the SCEP responder first validates the requestor's identity and proof of possession of a unique credential, before requesting the Issuing CA to issue a new certificate, forwarding the new certificate back to the SCEP client.

- The SCEP Responder can also reject the certificate request, or indicate issuance is pending based on an Issuing CA action. On receipt of a replacement certificate chain the *device automation client* validates the certificate chain received including testing against either CRL or OCSP responses. Only if the new certificate chain is known to be good will the certificate chain be written to the application certificate store, overwriting the previous certificate. On renewal, a peer's trust anchor(s) can also be renewed.

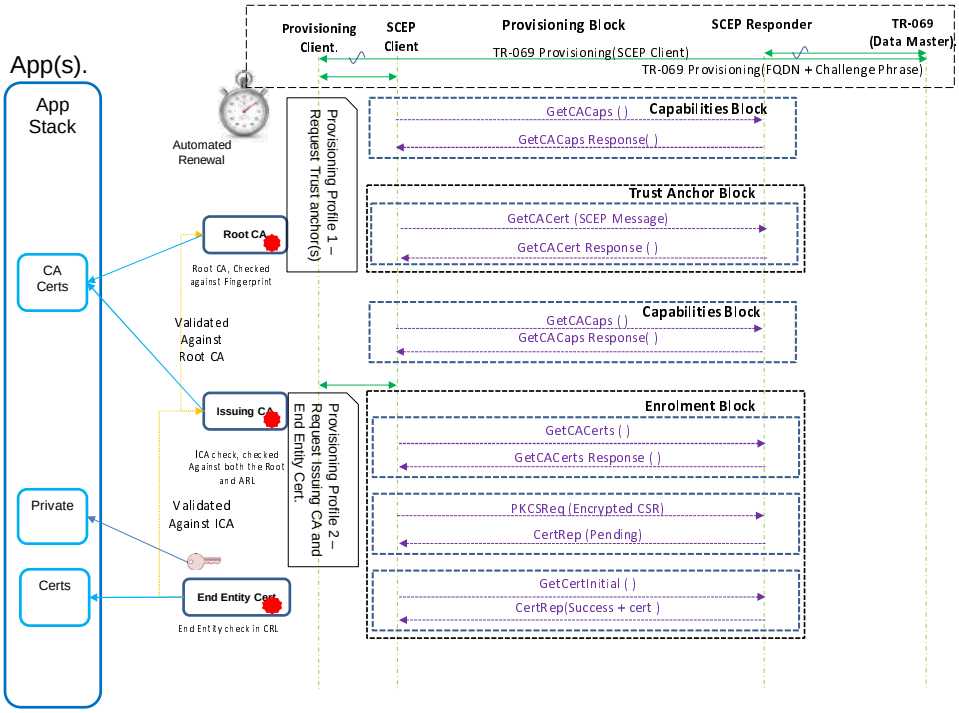


Figure M.2-1: SCEP certificate provisioning procedure

The SCEP certificate automation solution consists of the following five functions:

1) Initial configuration of the SCEP client with Provisioning Profiles

Initial configuration of the SCEP client addresses the need to establish sets of context specific Provisioning Profiles within an end point device. The two obvious options for providing Provisioning Profiles are:

1) Manual configuration of each device; and

2) Automated provisioning from a device manager or element manager service. For example by the procedures in ETSI TS 118 122 [57].

The number of sets of Provisioning Profiles matches the number of Application Security Stacks required.

This function downloads a set of Provisioning Profiles from the device manager or element manager service to enable the following actions:

- A unique x509v3 cryptographic credential chaining to a trusted Root CA is established. allowing the end point device to subsequently bootstrap its setup.
- A locally significant unique key pair is established.
- An associated certificate signing request is generated.
- A trust anchor is validated out of band by verification of a finger print within the Provisioning Profile.
- Each subordinate CA retrieved is validated in turn against its superior.
- The request of a client certificate from a pre-authorized issuer (the SCEP responder) is authenticated and secured using a username and password.
- The trust anchor of a trusted peer can also be downloaded and validated. These peer trust anchors can be updated based on a revised Provisioning Profile.

2) Device Intelligence & State Machine

The device intelligence and state machine is the heart of any SCEP, Certificate Management Protocol version 2 (CMPv2) or EST solution. Logically a good state machine can drive any message responder where SCEP is considered here.

The state machine is triggered by a complete and valid set of Provisioning Profiles.

This function, while intended to operate autonomously in the context of unattended IoT devices without a web browser user interface, has been written to reflect a browser based user journey. The intention is to maintain compatibility with any manual test and diagnostic processes required for IoT devices and with elements of the service that do have a traditional user interface, for example, use of a smart phone in the oneM2M Home Domain.

The key steps are:

1) The device requests its own Trust Anchor (Root CA).

2) The device's own Trust Anchor (Root CA) is validated against a fingerprint provided by a Provisioning Profile.

3) The device requests its own intermediate certificates one by one.

4) The device intermediate certificates are validated against the superior issuer to protect against MITMA.

5) The device requests a first client certificate. This assumes a device has no client certificate, but is in possession of a valid set of Provisioning Profiles. This step always requests the issuing CA to provide confidentiality for certificate requests. The SCEP client recovers the public key for the ICA. The certificate request is encrypted with the public key so that only the CA or RA private key can decrypt the request.

6) If directed by provisioning authority, the device requests a new certificate, or requests the renewal of an existing certificate immediately. A request for a new certificate might be against a different PKI.

7) Automated renewal of an existing certificate, based for example on a configured percentage of the current certificates lifetime has elapsed, is also supported.

8) All certificates are parsed to request associated CRLs or OCSP response.

9) The client requests the peers Trust Anchor, if it is different from its own Trust Anchor.

10) The intermediate and issuing CA of a peer are requested to allow mutual authentication if required. Once a new, or replacement, certificate chain has been established, the certificate chain is validated, as it will likely be used to replace the existing good certificate chain.

11) The key material is moved to the appropriate secure application stores.

12) The provisioning authority of current certificate is notified with information as required.

13) Expired certificate artefacts are deleted.

Note that the above list is not meant to imply a state machine order, or indicate a solution. However, it is assumed sophisticated solutions will exceed the states identified, and simpler solutions can choose to omit states not required by the device solution.

3) SCEP Client

A SCEP client is typically an open source piece of software developed to perform certificate request actions against the SCEP responder. The SCEP Client is directed by the state machine described above using the data provisioned in the initial configuration procedure.

The SCEP client is compliant with [66] and can be sourced from the open source communities, if a native client does not exist today. For example see [i.26] - based on original work by Martin Bartosch.

This SCEP Client was selected because the authors have modified their SCEP client behaviour to support long chain PKI (see [i.28]).

An alternative is the Java-based SCEP client at [i.27] by Dave Grant and team.

NOTE: This has also been modified to support long chained PKI and recently forked to specifically address Android requirements by Wes Bunton.

4) SCEP Responder

A SCEP responder is an additional component of both Enterprise and Managed PKI services. Essentially a SCEP responder can be considered as an additional RA (Registration Authority) service.

On request the SCEP responder(s) will provide Trust Anchors, Intermediate CAs, issuing CAs and Locally Significant certificates. A private/public key pair needs to be generated on the device.

Requests for certificate issuance would be against a unique username and password held securely within the request Subject Alternative Name and challenge phrase fields of the certificate CSR (see IETF RFC 8894 [66]).

Typically these one-time passwords expire on certificate issuance, needing to be re-set in the future when certificate renewal services are required.

The provisioning solution identified would be authoritative - tracking devices and elements under management, and would pre-provision the SCEP responder with valid username and password pairs, prior to the SCEP client using them.

Unsuccessful authentications are rejected, and successfully authenticated CSR are passed to the PKI for fulfilment. Upon successful authentication, an End Entity Certificate is returned.

The provisioning solution can even request revocation of device certificates that can no longer be trusted.

5) Locally Significant PKI & Certificates

A PKI service provides the pre-requisite knowledge, skill and Compliance Framework to support SCEP certificate issuance. The building blocks of a SCEP solution include: PKI&CA, SCEP Responder (RA), Request Authenticator, and Request Authorizer.

It is typical in the CPE or IoT space that a PKI is designed based on a good understanding of the certificate volumes, and an understanding of the required cryptographic operational separation to be enforced.

Certificate Authority

A SCEP Certification Authority (CA) signs client certificates. The CAs name is stored in the issuer field of resulting certificates. Before any PKI operations are invoked, the SCEP responder shares an issuer 'CA' certificate that is compliant with the profile in IETF RFC 5280 [34] with SCEP Client and optionally dedicated RA certificates. This can be a CA certificate that was issued by a higher level CA. The client builds an entire certificate chain from the trust anchor, validating each certificate in turn.

Registration Authority

A SCEP Registration Authority (RA) as a SCEP Responder performs validation and authorization checks of the SCEP requester and forward the certification requests to the CA. The SCEP Responder receives a certificate from the CA and forwards this to the SCEP Client. The RAs name does not appear in the issuer field of resulting certificates.

Requester Authentication

As with every protocol that uses public-key cryptography, the association between the public keys used in the protocol and the identities with which they are associated are authenticated in a cryptographically secure manner. This requirement is needed to prevent a "man-in-the-middle" attack, in which an adversary can manipulate the data as it travels between the protocol participants and subvert the security of the protocol. The communication between the requester and the certification authority is secured using SCEP Secure Message Objects which specifies how PKCS#7 is used to encrypt and sign the data of the CSR.

Request Authorization

The following SCEP authentication methods for certificate authorization can be supported:

- use of unique usernames and passwords;
- use of unique end entity certificate and a demonstration of proof of possession of the private key.

Annex N (informative): Considerations on Long Term Key Storage

Long term Provisioned Secure Connection Keys can pose a security risk if not adequately secured, and for this reason Long Term Provisioned Secure Connection Keys are recommended to be stored in Secure Environments.

Long term Pre-Provisioned Symmetric Enrolee Keys can pose a security risk if not adequately secured, and for this reason it is recommended that Long term Pre-Provisioned Symmetric Enrolee Keys are stored in Secure Environments.

Since by definition, there will be a very wide range of non-3GPP based devices with many different implementations. As these are not likely to be standardized, this is not in the scope of oneM2M or 3GPP. However, security best practice guides are available from:

- ETSI TS 103 645 - Cyber Security For Consumer Internet Of Things [i.33];
- IoTSF - Secure Design: Best Practice Guide. Release 2, November 2019 [i.34]; and
- GSMA - IoT Security Guidelines and Assessment [i.35].

Annex O (informative): Bibliography

- Open Mobile API specification V3.2.
- GlobalPlatform Device Technology TEE Client API Specification, Version 1.0.
- 3GPP TS 33.222: "Generic Authentication Architecture (GAA), Access to network application functions using Hypertext Transfer Protocol over Transport Layer Security (HTTPS) (Release 12)".
- 3GPP TS 24.109: "Bootstrapping interface (Ub) and network application function interface (Ua); Protocol details (Release 12)".
- 3GPP TS 29.109: "Protocols details Generic Authentication Architecture (GAA); Zh and Zn Interfaces based on Diameter protocol; Stage 3 (Release 12)".
- ISO/IEC 7816-6: "Identification cards - Integrated circuit cards - Part 6: Interindustry data elements".
- ISO/IEC 7816-8: "Identification cards - Integrated circuit cards - Part 8: Security related interindustry commands".
- ISO/IEC 7816-9: "Identification cards - Integrated circuit cards - Part 9: Additional interindustry commands and security attributes".
- GlobalPlatform Device Technology, Generic API to access Secure Elements, Open Mobila API Specifications, Version 3.2.
- ETSI TS 102 600: "Smart Cards; UICC - Terminal Interface; Characteristics of the USB Interface".
- ETSI TS 102 622: "Smart Cards; UICC - Contactless Front-End (CLF) Interface; Host Controller Interface".
- IEEE P1363™: "Standard Specifications for Public Key Cryptography".

History

Version	Date	Status
V4.7.1	March 2026	Publication