# M2M Security

國立交通大學資訊工程系
Department of Computer Science
National Chiao Tung University
November 22, 2018

## Outline

- Challenges in M2M Security
- oneM2M Security Solutions
- Access-rights Management
- Appendix. TLS/DTLS

# CHALLENGES IN M2M SECURITY

# Uniqueness of M2M Security (1)

- Security is an important function in any communication systems.
- This is especially critical in M2M systems because unlike in cellular systems, the application, service, network and device distribution are not tightly coupled and managed by a single entity such as a network operator.
- For example, in the case of smart meters, the entities involved may include
  - The utility company (M2M application provider)
  - The M2M service provider (MVNO providing M2M service layer)
  - The cellular network provider (providing 3G/4G networks)
  - The meter manufacturer (providing smart meter) and
  - The end user
- Furthermore, limited business relationships exist among these multiple players and thus no mutual trust should be assumed among these players.

# Uniqueness of M2M Security (2)

- The economics of M2M do not allow a security provisioning process similar to that of cellular handsets. Security methods developed for M2M should be able to deal with an explosion in the number of devices by manufacturers.
  - A simpler automated process with the capability of efficiently dealing with a large number of devices is desirable.
- Unlike cellular handsets, M2M devices are often unattended and are subject to a higher risk of vandalism and misuse.
  - Security solutions should be designed to protect against any inappropriate use of the M2M devices.

# Examples of Undetected M2M Security Attacks

- Theft of Removable UICC (Universal Integrated Circuit Card) from an M2M device
  - The network cannot tell that the UICC has been installed in a non-M2M device.
- Hijacking through Passive Medium Overhearing with the absence of UICC
  - By overhearing the UATI (Unicast Access Terminal Identifier) and the IP addresses that are assigned to the M2M devices by the network operators, the attacker can generate packet headers that will mislead the network to believe that the packets are from a legitimate client.

# Trust Relationships in the M2M Ecosystem

- The ecosystem potentially involves M2M application providers, M2M service providers, cellular network providers, device manufacturers and end users.
- Application providers may not have a relationship with device manufacturers.
- However, M2M application providers always maintain a business relationship with end users and M2M service providers; M2M service providers always maintain relationship with network providers.
- It is likely that the device may be used by an end user who has no ownership of the device.
- Examples: (1) smart meters offered by a utility company and (2) OBU (On Board Unit) offered by an M2M application provider for fleet management.

# M2M Security Threats

- Customer/M2M Device User
  - Devices access by unauthorized entities
  - Leaking of stored data by eavesdropping at some point of the network
  - End user identity revealed
- M2M Network Provider
  - Threat primarily from inexpensive devices and ease of accessibility of devices
- M2M Service Provider
  - Service availability
  - Data integrity
- Application Provider
  - Man-in-the middle attacks to the data transferred  from devices to back-end servers.
  - Non authenticated devices.

# M2M Security Requirements (1)

- Identification -the process of checking if the identity provided for authentication is valid
- Mutual Authentication - the process of validating if the identity supplied in the identification step is associated with a trustworthy credential.
- Authorization - responsible for authorizing services and data access to authenticated entities.
- Confidentiality – keep secrecy of the data transmitted.

# M2M Security Requirements (2)

- Integrity – Guarantee no modification of the original data
- Exclusivity - no use of M2M communication modules for non M2M applications.
- Anonymity – Keep the sender’s identify unrevealed.

# Bootstrapping Requirements

- Complex ecosystem: assume no trust relationship between application providers and device manufacturers.
- Scalability: Bootstrapping should be as automated as possible.
- Network authentication first: The M2M service layer is an overlay network over the existing network.
- Flexibility in operator selection: No leakage of key information across operator boundaries.

# Available Security Solutions for Bootstrapping

- Public Key (Asymmetric Key) Solutions
- Smart Card-based Solutions
- Pre-Provisioned Symmetric Key Solutions
- Identity-Based Encryption Solutions

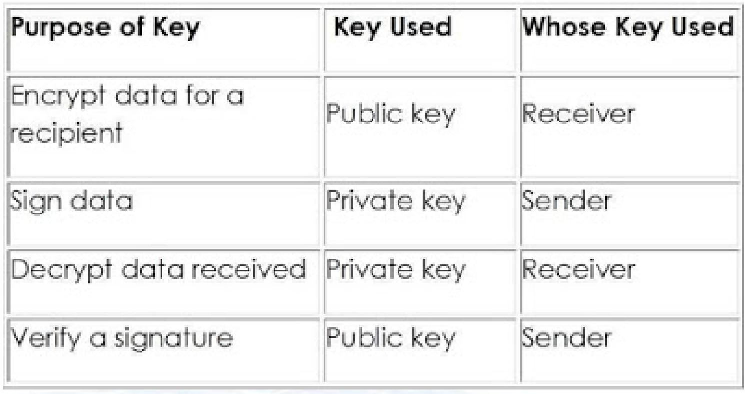
# Authenticated key

## Symmetric Key vs. Asymmetric Key

- Authenticated key agreements are cryptographic protocols where two or more participants authenticate each other and agree on key material used for securing future communication.
  - Symmetric key protocol
    - require an out-of-band security mechanism to bootstrap a secret key.
  - Asymmetric public-key protocol
    - require certificates and a large-scale Public Key Infrastructure (PKI). ← more flexible

# Public Key Solutions

- Approach: Provisioning a private-public key pair along with a **certificate of public key** during manufacturing.
- Public Key - provided by a CA (Certificate Authority) to encrypt data. Everyone usually knows public key.
- Private Key - a secret key known only to a key owner.

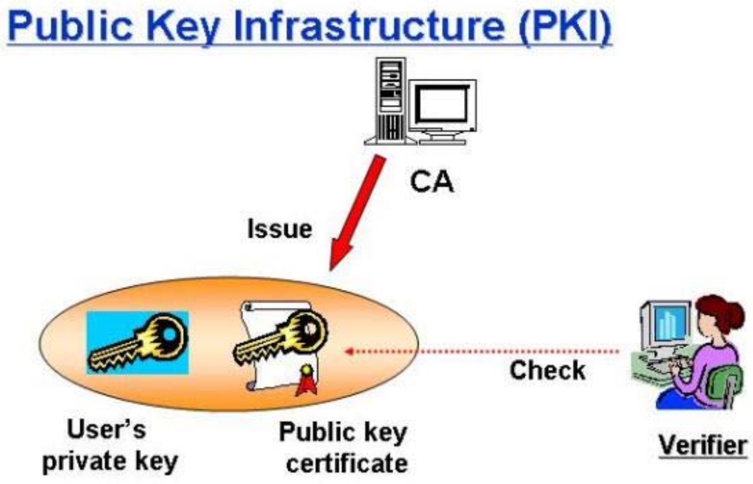


| Purpose of Key | Key Used | Whose Key Used |
|---|---|---|
| Encrypt data for a recipient | Public key | Receiver |
| Sign data | Private key | Sender |
| Decrypt data received | Private key | Receiver |
| Verify a signature | Public key | Sender |

# Public Key Solutions (Cont.)

- A **public key certificate** allows the exchange of information securely over the Internet using the public key infrastructure (PKI).
- A **certificate** was issued by an official, trusted agency called Certificate  Authority (CA) so that a recipient can verify that the certificate is real.
- To verify a **certificate** is genuine and valid, it is digitally signed by a **root certificate** belonging to the CA.
- Issues:
  - Need a large-scale PKI to handle billions of devices.
  - Must trust the CA (Trusted Third Party – TTP).
- PKI Standards: X.509

# Public Key Solutions (Cont.)



Source: http://ei4africa.eu/2013/04/24/ei4africa-boosts-the-deployment-of-public-key-infrastructures-in-africa/

# Public Key Solutions (Cont.)

- Why not ideal for M2M devices:
  - Public key methods are not supported by existing cellular networks. Existing networks use symmetric key protocols.
  - Cost of PKI is too high – one certificate per device means billions of certificates.
  - Safe storage of certificates and private keys is an issue.

# Smart Card-based Solutions

- Approach: Devices are equipped with smart cards to provide the necessary key material for use by the M2M service layer.
- Smart cards contain a pre-provisioned secret key that is hard-coded by the smart-card manufacturer for network operators.
- As a result, M2M application and service providers need to trust network operators.
- Issues:
  - Not scalable in cellular M2M deployments that may be comprised of billions of devices.
  - Subscription portability enabled by smart cards has little and no value in the M2M context but introduces the possibility of malicious use of these smart cards in other devices.
  - When switch network operators, all smart cards for an M2M application provider need to be replaced.

## Pre-Provisioned Symmetric Keys Solutions

- Approach: Both the M2M device and the service provider are pre-provisioned with the same key for symmetric key operations during device registration.
- Issues:
  - The M2M service provider and the device owner normally are not involved in the device deployment and distribution.
  - Manufacturers have to install the key into the device in the factory and thus they need to provide this information to the service provider.
  - Such pre-provisioned keys cannot be used as permanent keys for establishing secure service-layer sessions. However, they can be used for initial mutual authentication between devices and service providers and for use in a secure procedure to generate a new shared permanent key that cannot be derived by untrusted parties such as manufacturers.

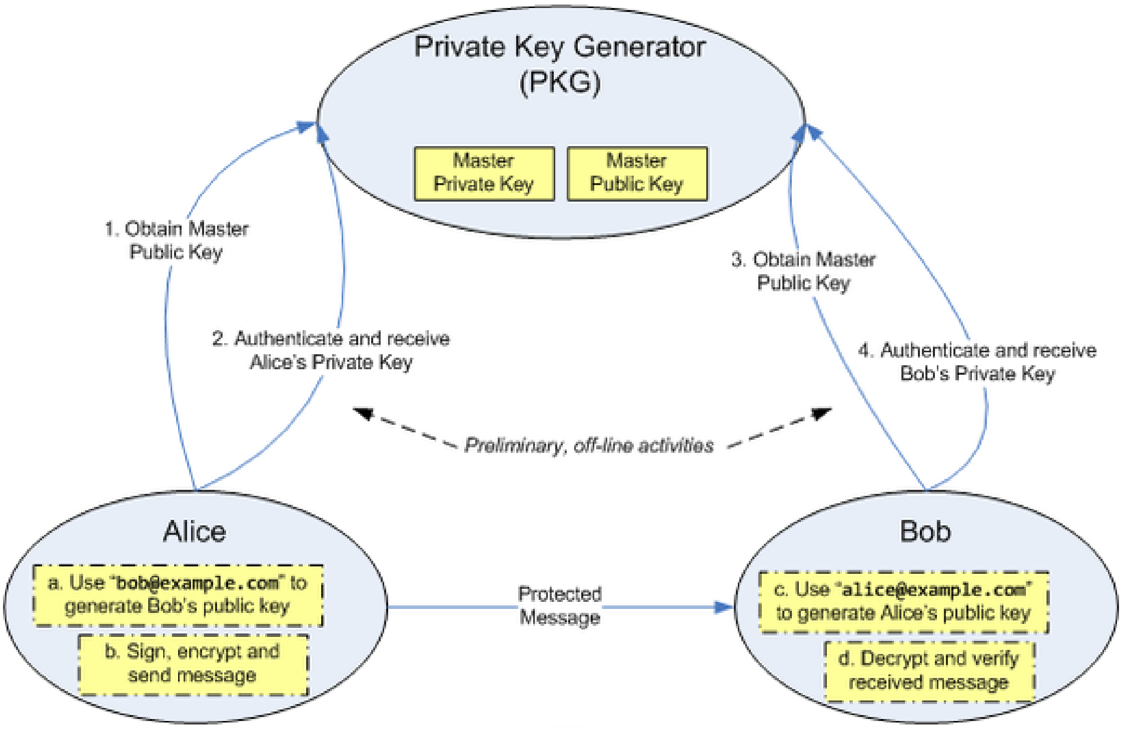
## Identity-Based Encryption Solutions

- Recently, **Identity-Based Encryption (IBE)** protocols have been proposed as a viable alternative to public-key methods by replacing the PKI with a Private-Key Generator (PKG).
- Approach: The public key of a user-device is a mathematical function of the identity that is associated with this key. Thus there is no need to bind an identity of an entity with a public key through the use of **certificates**. The PKG is used to generate the paired private key of the identity-based public key.

# Identity-Based Encryption Solutions (Cont.)

- IBE is based on the concept of **elliptic curve cryptography (ECC)** and can be used for establishing **a shared key** between two entities.
- Advantages:
  - keys are always valid (no key revocation.)
  - Public keys are derived from identifiers, thus the need for a public key infrastructure is eliminated.
  - Additional information may be encoded into the identifier such as an expiration date for a message.
- Drawbacks:
  - Key escrow problem: All private keys needed to decrypt encrypted data are held in escrow by PKG.
  - Lack of non-repudiation: Because PKG generates private keys for users, it may decrypt and/or sign any message without authorization.
  - A secure channel between a user and the PKG is required for transmitting the private key.

# Identity-Based Encryption Solutions (Cont.)



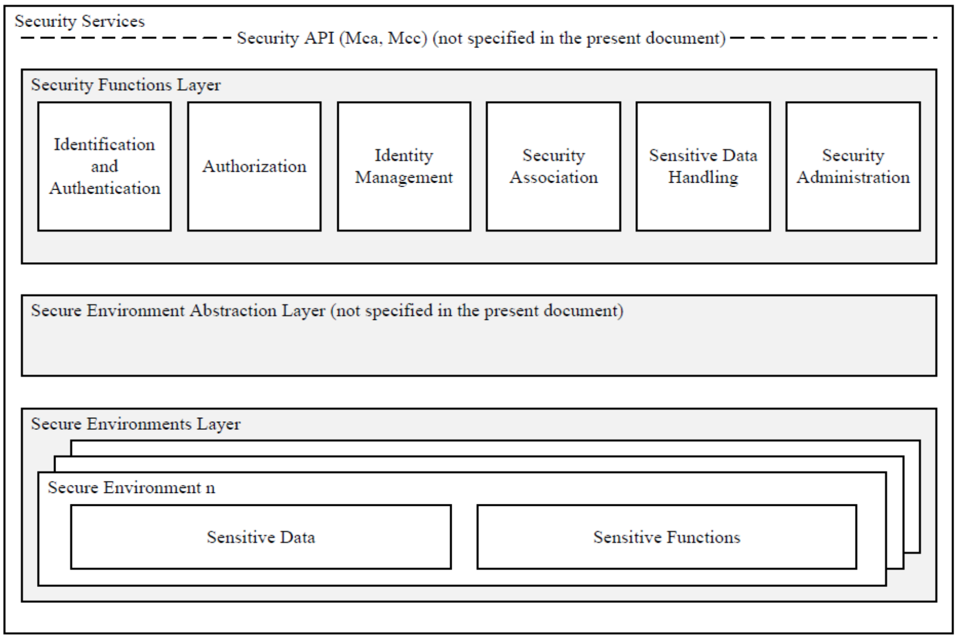
Source: http://en.wikipedia.org/wiki/ID-based_encryption

# ONEM2M SECURITY SOLUTIONS

## M2M Service Security Management

- oneM2M Security Architecture
- High Level Flow To Establish M2M Connection
  - M2M Service Layer Security Provisioning
  - M2M Security Association Establishment
- Detailed Flow To Establish M2M Connection
  - M2M Service Layer Security Provisioning
  - M2M Security Association Establishment

# oneM2M Security Architecture (1)



Security Services

Security API (Mca, Mcc) (not specified in the present document)

Security Functions Layer

Identification and Authentication

Authorization

Identity Management

Security Association

Sensitive Data Handling

Security Administration

Secure Environment Abstraction Layer (not specified in the present document)

Secure Environments Layer

Secure Environment n

Sensitive Data

Sensitive Functions

**Figure 5.1-1: High level overview of the Security architecture**

Source: oneM2M TS-0003-Security-Solutions

# oneM2M Security Architecture (2)

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.
    - Identification
    - Authentication
    - Authorization
    - Security Association
    - Sensitive Data Handling and
    - Security Administration.

Source: oneM2M TS-0003-Security-Solutions

# oneM2M Security Architecture (3)

- **Security Environment Abstraction Layer**:
  - Includes various security capabilities such as
    - key derivation,
    - data encryption/decryption,
    - signature generation/verification,
    - security credential read/write from/to the Secure Environments
  - The security functions in the Security Functions Layer invoke these functions in order to do the operations related to 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 not specified in the initial release but is expected to be considered in future releases.

Source: oneM2M TS-0003-Security-Solutions

# oneM2M Security Architecture (4)

- **Secure Environment (SE) layer**:
  - This layer contains one or multiple secure environments that provide various security services related to sensitive data storage and sensitive function execution.
  - The sensitive data includes SE capability, security keys, local credentials, security policies, identity information, subscription information, and so on.
  - The sensitive functions include data encryption, data decryption, and so on.
  - Implementation of secure environments is out of scope of the present document.

Source: oneM2M TS-0003-Security-Solutions

# Security Function Layer (1)

This layer contains a set of security functions that are exposed at Reference Points Mca and Mcc.

1. **Identification**
   - Check if the identity provided for authentication is valid.
   - Once passing the identification, the identified identity will be supplied to authentication process.
2. **Authentication**
   - Validate if the identity supplied in the identification step is associated with a trustworthy credential.
3. **Authorization**
   - Authorize services and data access to authenticated entities according to provisioned **Access Control Policies (ACPs)** and assigned roles.
   - The Authorization function may need to evaluate multiple ACPs in an authorization process.

# Security Function Layer (2)

4. **Security Association**
   - Establish the secure association between two M2M nodes to provide confidentiality and integrity.
5. **Sensitive Data Handling**
   - Protect the sensitive data like local credentials for storage and manipulation.
   - Perform security algorithms in cryptographically separated secure environments.
6. **Security Administration**
   - Provide functions to manage the Security functions, resources and attributes.
   - Manage sensitive data with their associated identifiers and subscriptions on behalf of other entities.

# High Level Flow To Establish M2M Connection

1. First, establish connectivity in the underlying Network Services Layer
2. Next, **Enrollment Phase: Service Layer Security Provisioning**
3. Next, **Operational Phase: Security Association Establishment**
4. **TLS or DTLS session** established which protects messages being exchanged between adjacent AE/CSE, i.e. hop-by-hop.
5. Finally, AEs that need to preserve the privacy of their information exchange from untrusted intermediate nodes may be provisioned to support a direct **security association** between them.

# Objectives in Each Stage

- RSPF (Enrollment Key, Master Credentials)
- SAEF (Security Association)
- Between Applications (Security Association)

# Enrollment Phase: Service Layer Security provisioning

- May 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 Framework RSPF**) that takes place before the equipment starts actual operation.
- Requires selection of the stakeholder that will provide services through the equipment, especially the M2M Service Provider

# Remote Security Provisioning Frameworks (RSPF)

- Remote Security Provisioning procedures rely on an **M2M Enrollment Function (MEF)** which can be external to the M2M Service Provider to establish appropriate credentials.
- Provide post-provisioning of the essential information (normally to produce an **Enrollment Key**, then generate **Master Credentials** from Enrollment Key) to establish a security association between a Field Domain entity and a chosen M2M Service Provider.

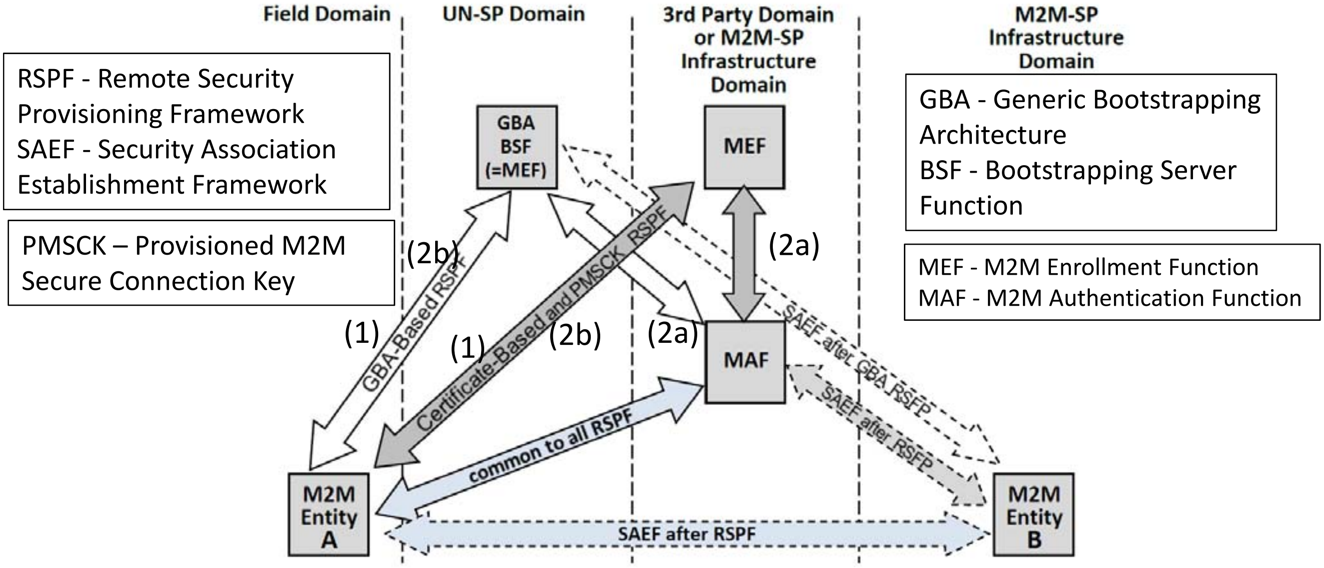
# Operational Phase: Security Association Establishment

- 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.
- This mutual authentication provides protection from unauthorized access and Denial of Service attacks. It also enables 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.

# Security Association Establishment Framework (SAEF)

- On the Mca and Mcc reference points, **security association** establishment between a field domain CSE and an infrastructure 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 AEs and the CSE in the field domain is strongly recommended.

# Roles of GBA BSF and MEF (1)



RSPF - Remote Security Provisioning Framework
SAEF - Security Association Establishment Framework

PMSCK – Provisioned M2M Secure Connection Key

GBA - Generic Bootstrapping Architecture
BSF - Bootstrapping Server Function

MEF - M2M Enrollment Function
MAF - M2M Authentication Function

**Figure 6.1.2.1-1: Entities involved in Remote Security Provisioning**

Source: oneM2M TS-0003-Security-Solutions

# Roles of GBA BSF and MEF (2)

(1) Three alternatives between M2M Entity A and MEF for service bootstrapping

- Pre-Provisioned Symmetric Key RSPF (not shown in the diagram)
- Certificate-Based RSPF
- GBA-based RSPF

(2a) & (2b) Derived key by RSPF is distributed to MAF and M2M Entity A for MAF SAEF

# SAEF and the Role of MAF (1)

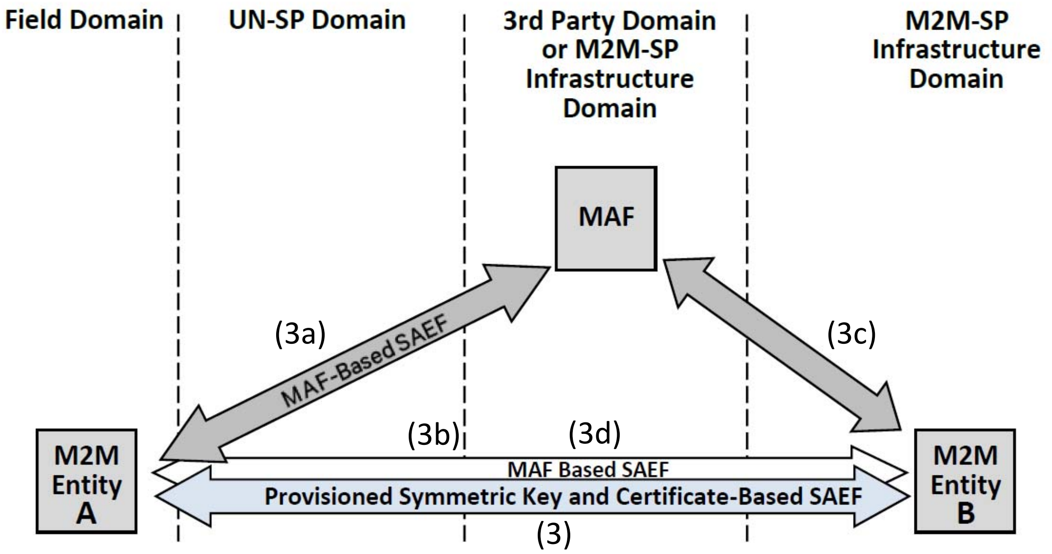

Field Domain
UN-SP Domain
3rd Party Domain or M2M-SP Infrastructure Domain
M2M-SP Infrastructure Domain
MAF
(3a)
MAF-Based SAEF
(3c)
(3b)
(3d)
M2M Entity A
M2M Entity B
MAF Based SAEF
Provisioned Symmetric Key and Certificate-Based SAEF
(3)

**Figure 6.1.2.2.1-1: Entities involved in Security Association Establishment**

Source: oneM2M TS-0003-Security-Solutions

# SAEF and the Role of MAF (2)

3) M2M Entity A establishes security association with M2M Entity B based on

- Provisioned Symmetric Key SAEF (no MAF involved)
- Certificate-Based SAEF (no MAF involved)
- M2M Authentication Function (MAF) SAEF

For MAF SAEF,

(3a) Entity A handshakes with MAF

(3b) Entity A sends the derived Kc and KcID to Entity B

(3c) Entity B handshakes with MAF

(3d) Entity A and Entity B handshake to set up security association

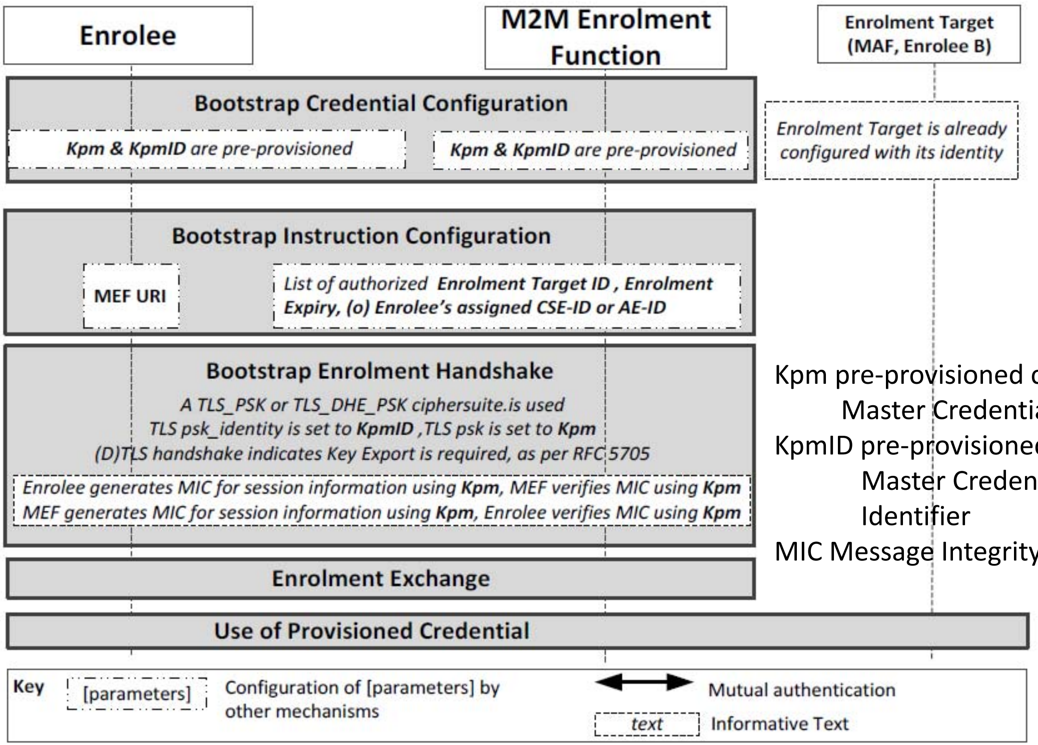
# Security Mechanism defined by oneM2M

| | Remote Security Provisioning Framework | Security Association Establishment Framework |
|---|---|---|
| 1 | Pre-provisioned Symmetric Key RSPF | Provisioned Symmetric Key SAEF |
| X | Pre-provisioned Symmetric Key RSPF | Certificate-based SAEF |
| 2 | Pre-provisioned Symmetric Key RSPF | MAF-Based SAEF |
| 3 | Certificate-based RSPF | Provisioned Symmetric Key SAEF |
| X | Certificate-based RSPF | Certificate-based SAEF |
| 4 | Certificate-based RSPF | MAF-based SAEF |
| 5 | GBA-based RSPF | Provisioned Symmetric Key SAEF |
| X | GBA-based RSPF | Certificate-based SAEF |
| 6 | GBA-based RSPF | MAF-based SAEF |
| 7<br>8<br>9 | ----- | Provisioned Symmetric Key SAEF<br>MAF-based SAEF<br>Certificate-based SAEF |

# List of Keys

- Kc M2M Secure Connection Key
- KcID M2M Secure Connection Key Identifier
- Ke Enrolment Key
- KeID Enrolment Key Identifier
- Ker Enrolment Re-Authentication Key
- Km Master Credential
- KmID Master Credential Identifier
- Kpm pre-provisioned credential for Master Credential provisioning
- KpmID pre-provisioned credential for Master Credential provisioning Identifier
- Kpsa provisioned credential for M2M Security Association Establishment
- KpsaID provisioned credential for M2M Security Association Establishment Identifier
- Ks temporary Key material referred to in GBA
- Ks..NAF Abbreviation of Ks_(int/ext)_NAF
- Ks_(ext/int)_NAF Derived key in GBA_ME or Derived key in GBA_U which remains on UICC
- Ks_ext_NAF Derived key in GBA_U sent to the ME
- Ks_int_NAF Derived key in GBA_U which remains on UICC
- Ks_NAF Derived key in the ME

# RSP based on Pre-Provisioned Symmetric Key



Enrolee

M2M Enrolment Function

Enrolment Target (MAF, Enrolee B)

**Bootstrap Credential Configuration**

*Kpm & KpmID are pre-provisioned*

*Kpm & KpmID are pre-provisioned*

*Enrolment Target is already configured with its identity*

**Bootstrap Instruction Configuration**

MEF URI

*List of authorized **Enrolment Target ID** , **Enrolment Expiry, (o) Enrolee's assigned CSE-ID or AE-ID***

**Bootstrap Enrolment Handshake**

*A TLS_PSK or TLS_DHE_PSK ciphersuite.is used*

*TLS psk_identity is set to **KpmID** ,TLS psk is set to **Kpm***

*(D)TLS handshake indicates Key Export is required, as per RFC 5705*

*Enrolee generates MIC for session information using **Kpm**, MEF verifies MIC using **Kpm***

*MEF generates MIC for session information using **Kpm**, Enrolee verifies MIC using **Kpm***

**Enrolment Exchange**

**Use of Provisioned Credential**

Key [parameters] Configuration of [parameters] by other mechanisms

Mutual authentication

*text* Informative Text

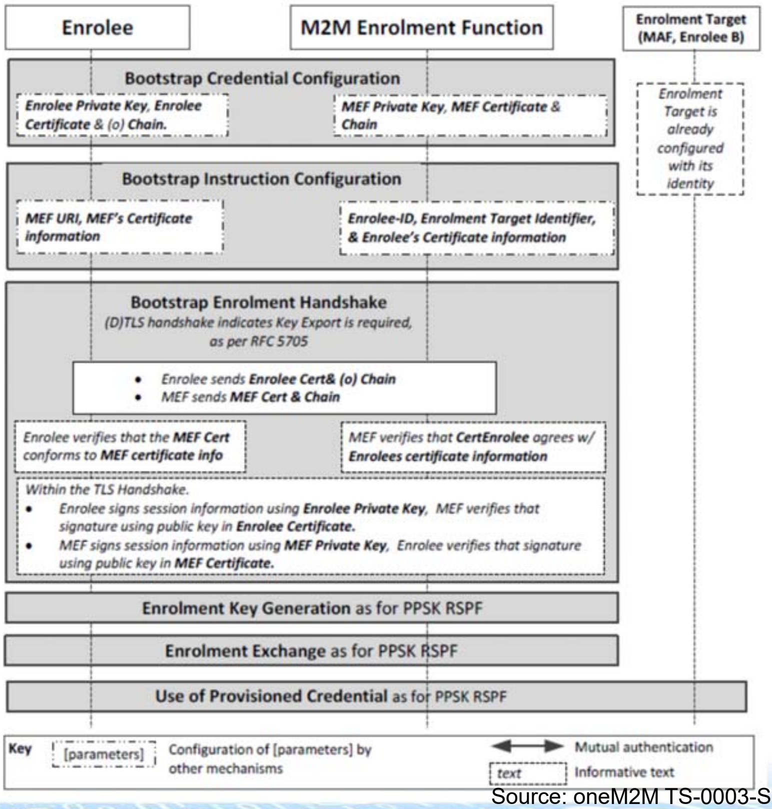
Kpm pre-provisioned credential for
Master Credential provisioning

KpmID pre-provisioned credential for
Master Credential provisioning
Identifier

MIC Message Integrity Code

Source: oneM2M TS-0003-Security-Solutions Release 2

# RSP based on Certificate



Enrolee

M2M Enrolment Function

Enrolment Target (MAF, Enrolee B)

**Bootstrap Credential Configuration**

***Enrolee Private Key, Enrolee Certificate** & (o) **Chain.***

***MEF Private Key, MEF Certificate** & **Chain***

*Enrolment Target is already configured with its identity*

**Bootstrap Instruction Configuration**

***MEF URI, MEF's Certificate information***

***Enrolee-ID, Enrolment Target Identifier, & Enrolee's Certificate information***

**Bootstrap Enrolment Handshake**

*(D)TLS handshake indicates Key Export is required, as per RFC 5705*

- *Enrolee sends **Enrolee Cert& (o) Chain***
- *MEF sends **MEF Cert & Chain***

*Enrolee verifies that the **MEF Cert** conforms to **MEF certificate info***

*MEF verifies that **CertEnrolee** agrees w/ **Enrolees certificate information***

*Within the TLS Handshake.*

- *Enrolee signs session information using **Enrolee Private Key**, MEF verifies that signature using public key in **Enrolee Certificate.***
- *MEF signs session information using **MEF Private Key**, Enrolee verifies that signature using public key in **MEF Certificate.***

**Enrolment Key Generation** as for PPSK RSPF

**Enrolment Exchange** as for PPSK RSPF

**Use of Provisioned Credential** as for PPSK RSPF

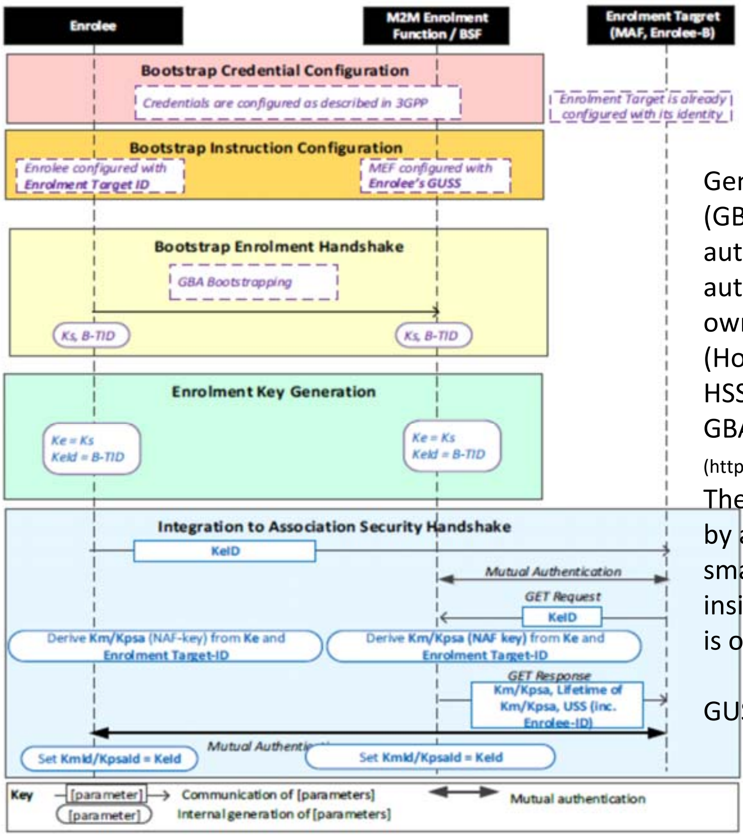
Key [parameters] Configuration of [parameters] by other mechanisms

Mutual authentication

*text* Informative text

Source: oneM2M TS-0003-Security-Solutions Release 2

# RSP based on GBA

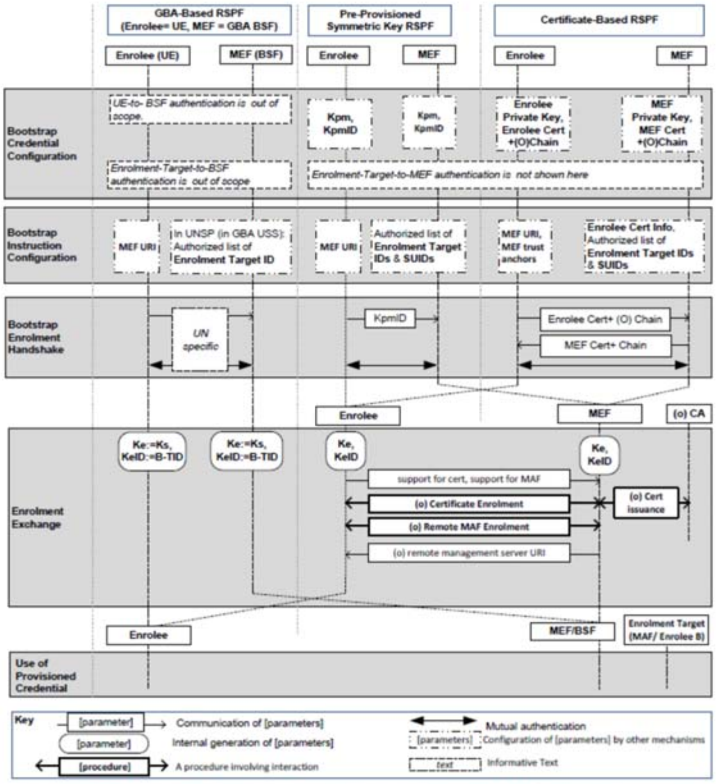

Enrolee
M2M Enrolment Function / BSF
Enrolment Targret (MAF, Enrolee-B)
Bootstrap Credential Configuration
Credentials are configured as described in 3GPP
Enrolment Target is already configured with its identity
Bootstrap Instruction Configuration
Enrolee configured with Enrolment Target ID
MEF configured with Enrolee's GUSS
Bootstrap Enrolment Handshake
GBA Bootstrapping
Ks, B-TID
Ks, B-TID
Enrolment Key Generation
Ke = Ks
KeId = B-TID
Ke = Ks
KeId = B-TID
Integration to Association Security Handshake
KeID
Mutual Authentication
GET Request
KeID
Derive Km/Kpsa (NAF-key) from Ke and Enrolment Target-ID
Derive Km/Kpsa (NAF key) from Ke and Enrolment Target-ID
GET Response
Km/Kpsa, Lifetime of Km/Kpsa, USS (inc. Enrolee-ID)
Mutual Authentication
Set Kmld/Kpsald = Keld
Set Kmld/Kpsald = Keld
Key
[parameter]
Communication of [parameters]
[parameter]
Internal generation of [parameters]
Mutual authentication

Generic Bootstrapping Architecture (GBA) is a technology that enables the authentication of a user. This authentication is possible if the user owns a valid identity on an HLR (Home Location Register) or on an HSS (Home Subscriber Server). GBA is standardized at the 3GPP (http://www.3gpp.org/ftp/Specs/html-info/33220.htm). The user authentication is instantiated by a shared secret, one in the smartcard, for example a SIM card inside the mobile phone and the other is on the HLR/HSS.

GUSS - GBA user security settings

Source: oneM2M TS-0003-Security-Solutions Release 2

# Detailed Flows of All Three RSP Methods


GBA-Based RSPF (Enrolee= UE, MEF = GBA BSF)
Pre-Provisioned Symmetric Key RSPF
Certificate-Based RSPF
Enrolee (UE)
MEF (BSF)
Enrolee
MEF
Enrolee
MEF
Bootstrap Credential Configuration
UE-to- BSF authentication is out of scope.
Kpm, KpmID
Kpm, KpmID
Enrolee Private Key, Enrolee Cert +(O)Chain
MEF Private Key, MEF Cert +(O)Chain
Enrolment-Target-to-BSF authentication is out of scope
Enrolment-Target-to-MEF authentication is not shown here
Bootstrap Instruction Configuration
MEF URI
In UNSP (in GBA USS): Authorized list of Enrolment Target ID
MEF URI
Authorized list of Enrolment Target IDs & SUIDs
MEF URI, MEF trust anchors
Enrolee Cert Info, Authorized list of Enrolment Target IDs & SUIDs
Bootstrap Enrolment Handshake
UN specific
KpmID
Enrolee Cert+ (O) Chain
MEF Cert+ Chain
Enrolee
MEF
(o) CA
Ke:=Ks, KeID:=B-TID
Ke:=Ks, KeID:=B-TID
Ke, KeID
Ke, KeID
Enrolment Exchange
support for cert, support for MAF
(o) Certificate Enrolment
(o) Cert issuance
(o) Remote MAF Enrolment
(o) remote management server URI
Enrolee
MEF/BSF
Enrolment Target (MAF/ Enrolee B)
Use of Provisioned Credential
Key
[parameter]
Communication of [parameters]
[parameter]
Internal generation of [parameters]
[procedure]
A procedure involving interaction
Mutual authentication
[parameters]
Configuration of [parameters] by other mechanisms
text
Informative Text

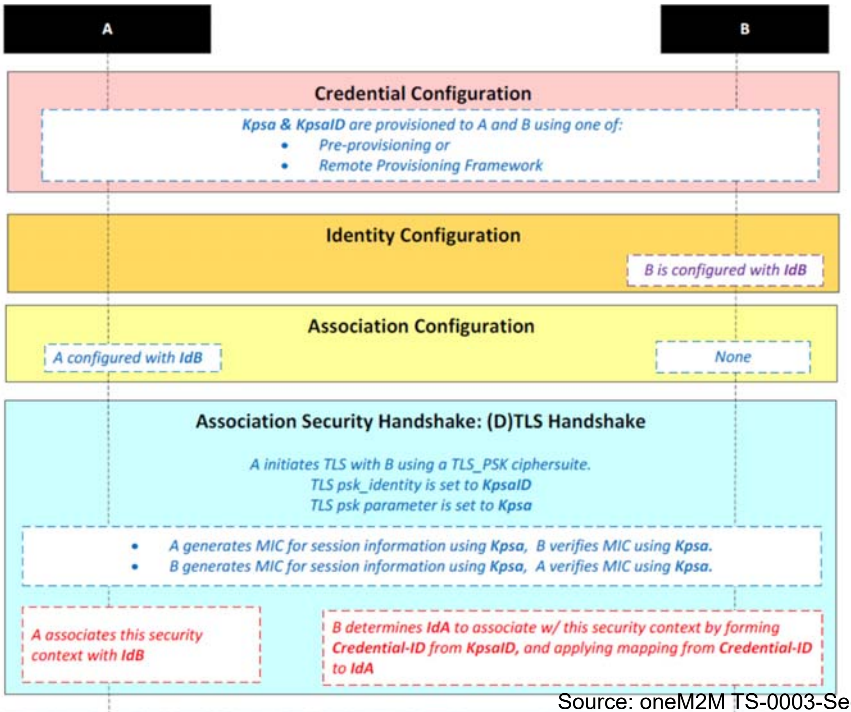
General Description of RSP

- Bootstrap Credential Configuration
- Bootstrap Instruction Configuration
- Bootstrap Enrolment Handshake
- Enrolment Key Generation
- Enrolment Phase

Figure 8.3.1.2-1: Overview of the Remote Security Provisioning Frameworks supported by oneM2M

Source: oneM2M TS-0003-Security-Solutions Release 2

# SAE Based on Provisioned Symmetric Key



A

B

Credential Configuration

*Kpsa & KpsaID are provisioned to A and B using one of:*
- *Pre-provisioning or*
- *Remote Provisioning Framework*

Identity Configuration

*B is configured with **IdB***

Association Configuration

*A configured with **IdB***

*None*

Association Security Handshake: (D)TLS Handshake

*A initiates TLS with B using a TLS_PSK ciphersuite.*
*TLS psk_identity is set to **KpsaID***
*TLS psk parameter is set to **Kpsa***

- *A generates MIC for session information using **Kpsa**, B verifies MIC using **Kpsa**.*
- *B generates MIC for session information using **Kpsa**, A verifies MIC using **Kpsa**.*

*A associates this security context with **IdB***

*B determines **IdA** to associate w/ this security context by forming **Credential-ID** from **KpsaID**, and applying mapping from **Credential-ID** to **IdA***

NOTE: Meaning of different font colors:

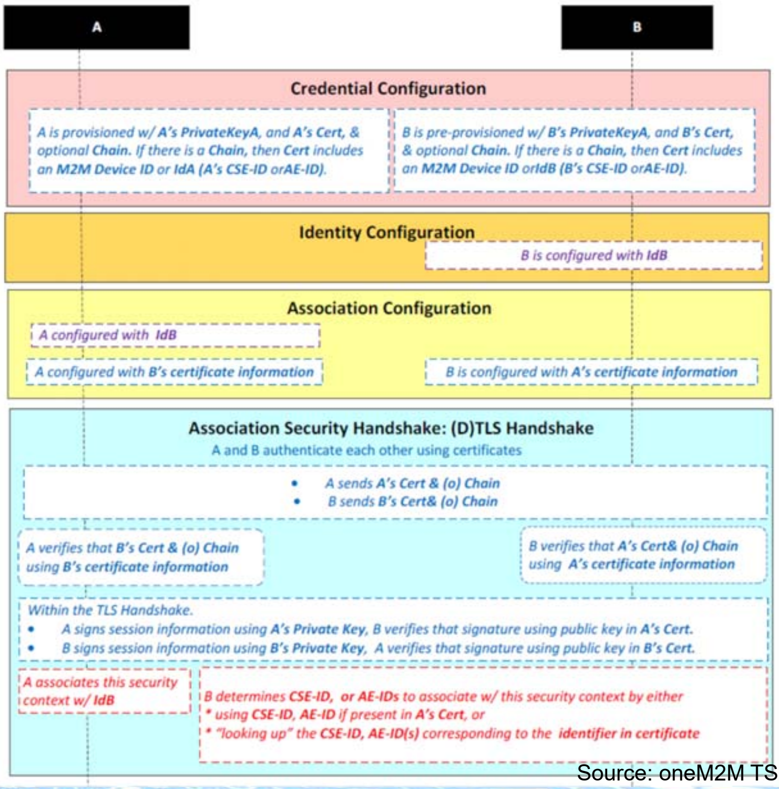
*Blue italic text highlights details specific to this particular Security Association Establishment Framework.*

*Purple italic text highlights technical actions that may include steps not specified by oneM2M.*

*Red italic text highlights security-related properties.*

Source: oneM2M TS-0003-Security-Solutions Release 2

# SAE Based on Certificate



A

B

Credential Configuration

*A is provisioned w/ **A's PrivateKeyA**, and **A's Cert**, & optional **Chain.** If there is a **Chain,** then **Cert** includes an **M2M Device ID** or **IdA (A's CSE-ID orAE-ID).***

*B is pre-provisioned w/ **B's PrivateKeyA**, and **B's Cert**, & optional **Chain.** If there is a **Chain,** then **Cert** includes an **M2M Device ID orIdB (B's CSE-ID orAE-ID).***

Identity Configuration

*B is configured with **IdB***

Association Configuration

*A configured with **IdB***

*A configured with **B's certificate information***

*B is configured with **A's certificate information***

Association Security Handshake: (D)TLS Handshake

A and B authenticate each other using certificates

- *A sends **A's Cert & (o) Chain***
- *B sends **B's Cert& (o) Chain***

*A verifies that **B's Cert & (o) Chain** using **B's certificate information***

*B verifies that **A's Cert& (o) Chain** using **A's certificate information***

*Within the TLS Handshake.*
- *A signs session information using **A's Private Key**, B verifies that signature using public key in **A's Cert.***
- *B signs session information using **B's Private Key**, A verifies that signature using public key in **B's Cert.***

*A associates this security context w/ **IdB***

*B determines **CSE-ID**, or **AE-IDs** to associate w/ this security context by either*
** using **CSE-ID, AE-ID** if present in **A's Cert**, or*
** "looking up" the **CSE-ID, AE-ID(s)** corresponding to the **identifier in certificate***

NOTE: Meaning of different font colors:

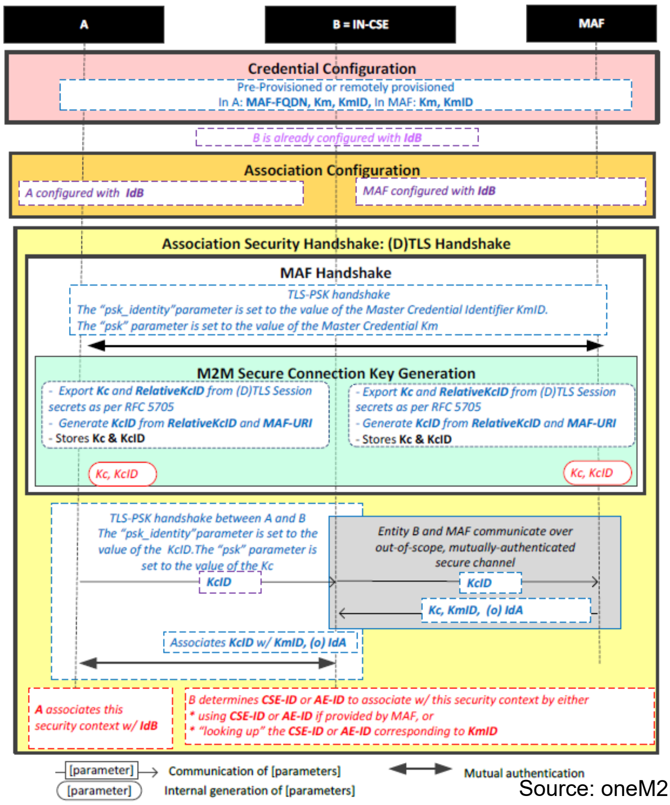
*Blue italic text highlights details specific to this particular Security Association Establishment Framework.*

*Purple italic text highlights technical actions that may include steps not specified by oneM2M.*

*Red italic text highlights security-related properties.*

Source: oneM2M TS-0003-Security-Solutions Release 2

# SAE Based on MAF



A

B = IN-CSE

MAF

**Credential Configuration**

*Pre-Provisioned or remotely provisioned*

In A: **MAF-FQDN, Km, KmID,** In MAF: **Km, KmID**

*B is already configured with **IdB***

**Association Configuration**

*A configured with **IdB***

*MAF configured with **IdB***

**Association Security Handshake: (D)TLS Handshake**

**MAF Handshake**

*TLS-PSK handshake*

*The "psk_identity"parameter is set to the value of the Master Credential Identifier KmID.*

*The "psk" parameter is set to the value of the Master Credential Km*

**M2M Secure Connection Key Generation**

- *Export **Kc** and **RelativeKcID** from (D)TLS Session secrets as per RFC 5705*
- *Generate **KcID** from **RelativeKcID** and **MAF-URI***
- Stores **Kc & KcID**

- *Export **Kc** and **RelativeKcID** from (D)TLS Session secrets as per RFC 5705*
- *Generate **KcID** from **RelativeKcID** and **MAF-URI***
- Stores **Kc & KcID**

*Kc, KcID*

*Kc, KcID*

*TLS-PSK handshake between A and B*

*The "psk_identity"parameter is set to the value of the KcID.The "psk" parameter is set to the value of the Kc*

***KcID***

*Entity B and MAF communicate over out-of-scope, mutually-authenticated secure channel*

***KcID***

***Kc, KmID, (o) IdA***

*Associates **KcID** w/ **KmID, (o) IdA***

***A** associates this security context w/ **IdB***

*B determines **CSE-ID** or **AE-ID** to associate w/ this security context by either*

** using **CSE-ID** or **AE-ID** if provided by MAF, or*

** "looking up" the **CSE-ID** or **AE-ID** corresponding to **KmID***

[parameter] Communication of [parameters]

[parameter] Internal generation of [parameters]

Mutual authentication

NOTE: Meaning of different font colors:

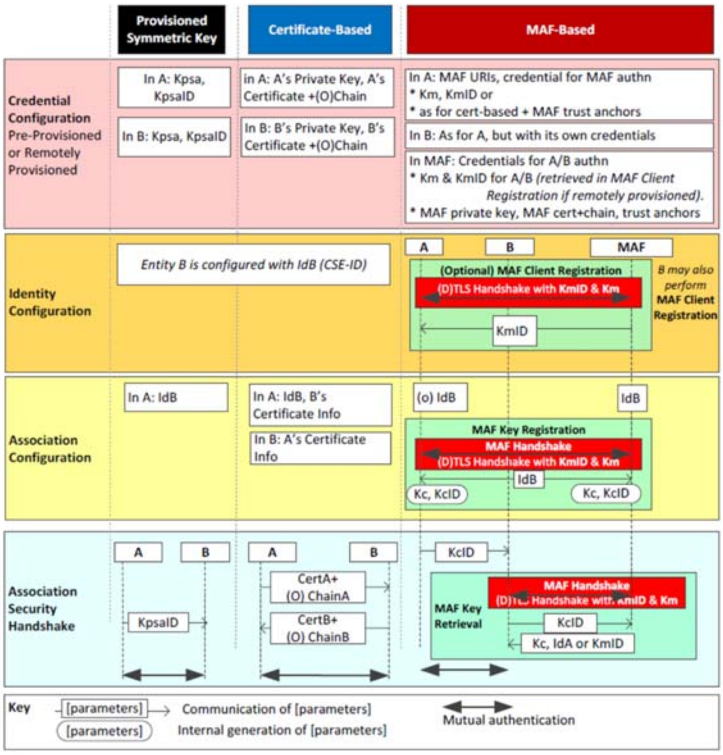
*Blue italic text highlights details specific to this particular Security Association Establishment Framework.*

*Purple italic text highlights technical actions that may include steps not specified by oneM2M.*

*Red italic text highlights security-related properties.*

Source: oneM2M TS-0003-Security-Solutions Release 1

# Detailed Flows of All Three SAE Methods



| | Provisioned Symmetric Key | Certificate-Based | MAF-Based |
|---|---|---|---|
| **Credential Configuration** Pre-Provisioned or Remotely Provisioned | In A: Kpsa, KpsaID<br>In B: Kpsa, KpsaID | in A: A's Private Key, A's Certificate +(O)Chain<br>In B: B's Private Key, B's Certificate +(O)Chain | In A: MAF URIs, credential for MAF authn<br>* Km, KmID or<br>* as for cert-based + MAF trust anchors<br>In B: As for A, but with its own credentials<br>In MAF: Credentials for A/B authn<br>* Km & KmID for A/B *(retrieved in MAF Client Registration if remotely provisioned).*<br>* MAF private key, MAF cert+chain, trust anchors |
| **Identity Configuration** | *Entity B is configured with IdB (CSE-ID)* | | A B MAF<br>(Optional) MAF Client Registration<br>(D)TLS Handshake with KmID & Km<br>KmID<br>*B may also perform* MAF Client Registration |
| **Association Configuration** | In A: IdB | In A: IdB, B's Certificate Info<br>In B: A's Certificate Info | (o) IdB<br>IdB<br>MAF Key Registration<br>MAF Handshake<br>(D)TLS Handshake with KmID & Km<br>IdB<br>Kc, KcID<br>Kc, KcID |
| **Association Security Handshake** | A B<br>KpsaID | A B<br>CertA+ (O) ChainA<br>CertB+ (O) ChainB | KcID<br>MAF Key Retrieval<br>MAF Handshake<br>(D)TLS Handshake with KmID & Km<br>KcID<br>Kc, IdA or KmID |
| **Key** | [parameters] Communication of [parameters]<br>[parameters] Internal generation of [parameters] | | Mutual authentication |

General description of SAEF

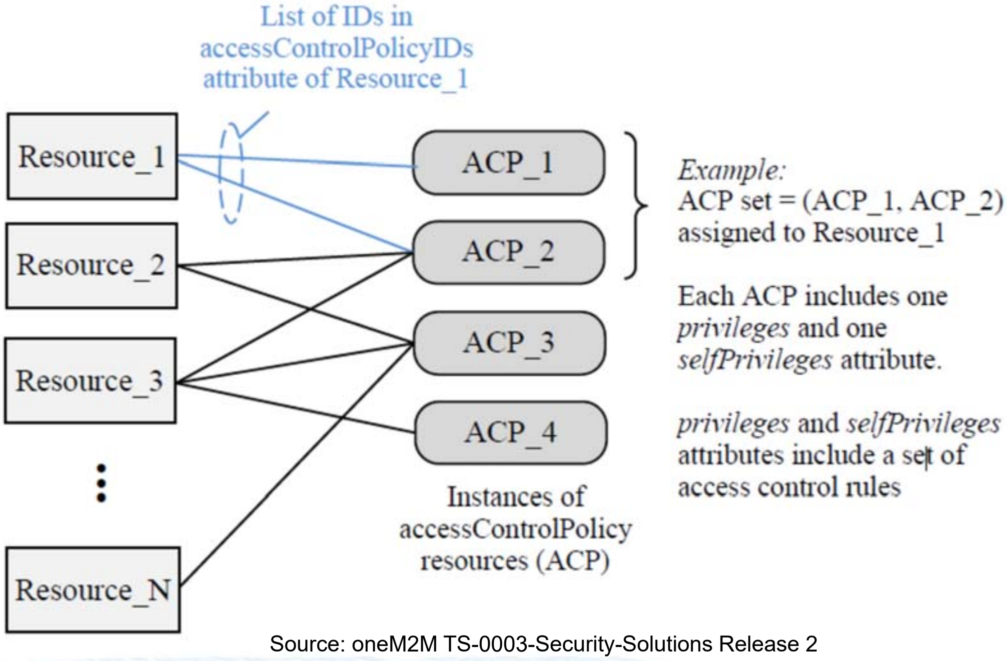
- Credential Configuration
- Identity Configuration
- Association Configuration
- Association Security Handshake

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

Source: oneM2M TS-0003-Security-Solutions Release 2

# ACCESS-RIGHTS MANAGEMENT

# Relation between Resource Instances and Access Control Policies


List of IDs in accessControlPolicyIDs attribute of Resource_1
Resource_1
Resource_2
Resource_3
⋮
Resource_N
ACP_1
ACP_2
ACP_3
ACP_4
Instances of accessControlPolicy resources (ACP)

*Example:*
ACP set = (ACP_1, ACP_2) assigned to Resource_1

Each ACP includes one *privileges* and one *selfPrivileges* attribute.

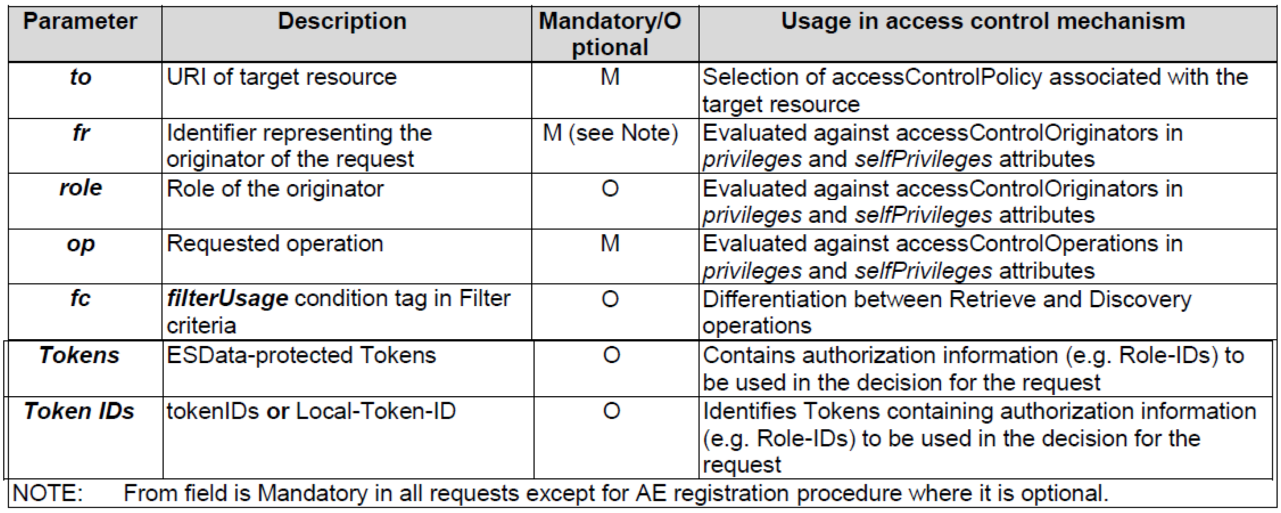
*privileges* and *selfPrivileges* attributes include a set of access control rules

Source: oneM2M TS-0003-Security-Solutions Release 2

# selfPrivileges vs. Privileges

- 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.

# 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 |
| ***fr*** | Identifier representing the originator of the request | M (see Note) | Evaluated against accessControlOriginators in *privileges* and *selfPrivileges* attributes |
| ***role*** | Role of the originator | O | Evaluated against accessControlOriginators in *privileges* and *selfPrivileges* attributes |
| ***op*** | Requested operation | M | Evaluated against accessControlOperations in *privileges* and *selfPrivileges* attributes |
| ***fc*** | ***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 |
| NOTE: From field is Mandatory in all requests except for AE registration procedure where it is optional. | | | |

Source: oneM2M TS-0003-Security-Solutions Release 2

# Condition for Permit Granted

- 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**.

# Access Control Rules

The set of access control rules is denoted as acrs.

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

Where

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

OR

*acr(k) = {acr(k)_accessControlOriginators, acr(k)_accessControlOperations, acr(k)_accessControlContexts}*

# ControlOriginators

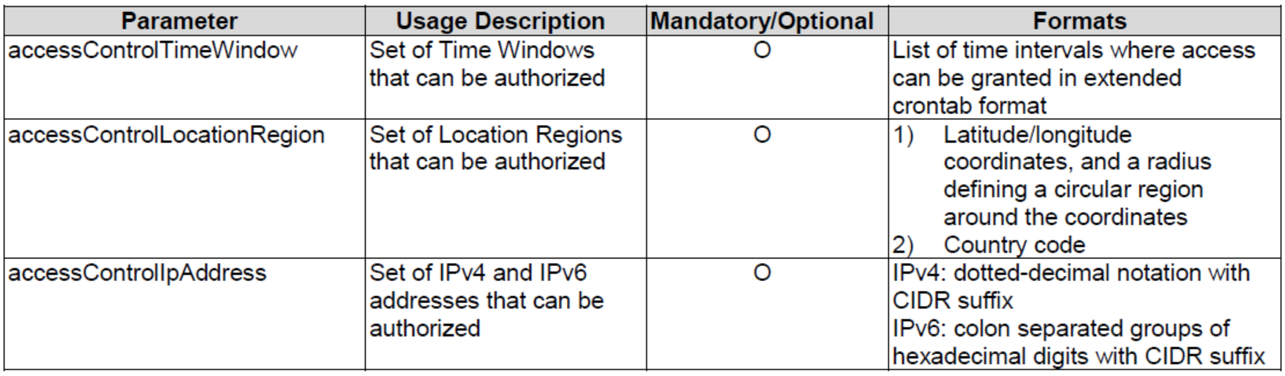
- The accessControlOriginators parameter comprises a list of CSE-IDs and/or AE-IDs of any format defined in oneM2M TS-0001.
- It is allowed to include the wildcard characters, e.g. "*", into the URI string of CSE-ID and AE-ID at any level.

# accessControlOperations

- The accessControlOperations parameter comprises a list of admissible operations which can be any subset of the following elements: Create, Request, Update, Delete, Discover, and Notify.
- While Create, Request, Update, Delete, and Notify operation are explicitly indicated in the op parameter of a request message.
- the Discovery operation is indicated by op = retrieve in combination with the provisioning of fc and Disrestype parameters in the request message.

# accessControlContexts

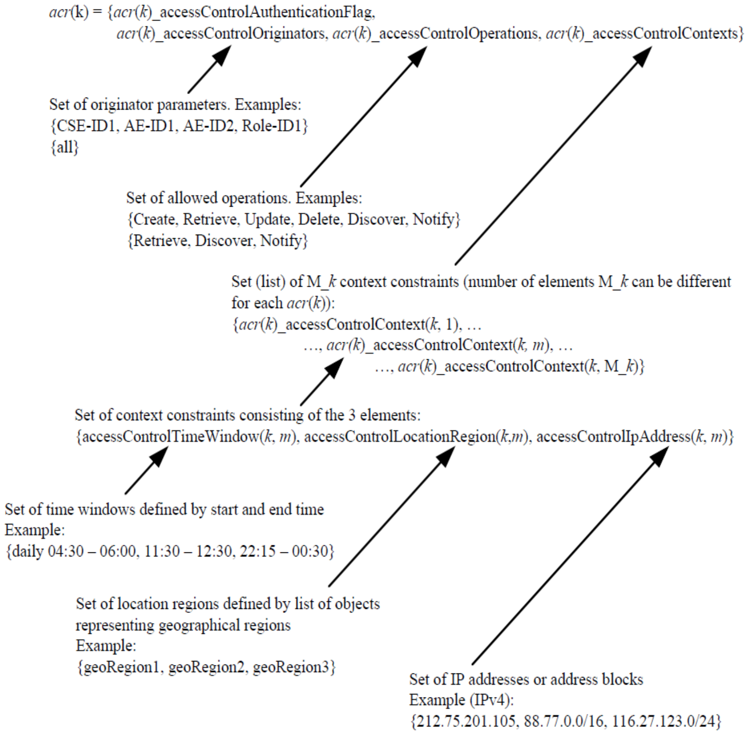
## Parameters of accessControlContexts



| 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<br>2) Country code |
| accessControlIpAddress | Set of IPv4 and IPv6 addresses that can be authorized | O | IPv4: dotted-decimal notation with CIDR suffix<br>IPv6: colon separated groups of hexadecimal digits with CIDR suffix |

Source: oneM2M TS-0003-Security-Solutions Release 2

# Access Control Decision



*acr*(k) = {*acr*(*k*)_accessControlAuthenticationFlag,
*acr*(*k*)_accessControlOriginators, *acr*(*k*)_accessControlOperations, *acr*(*k*)_accessControlContexts}

Set of originator parameters. Examples:
{CSE-ID1, AE-ID1, AE-ID2, Role-ID1}
{all}

Set of allowed operations. Examples:
{Create, Retrieve, Update, Delete, Discover, Notify}
{Retrieve, Discover, Notify}

Set (list) of M_*k* context constraints (number of elements M_*k* can be different for each *acr*(*k*)):
{*acr*(*k*)_accessControlContext(*k*, 1), …
…, *acr*(*k*)_accessControlContext(*k*, *m*), …
…, *acr*(*k*)_accessControlContext(*k*, M_*k*)}

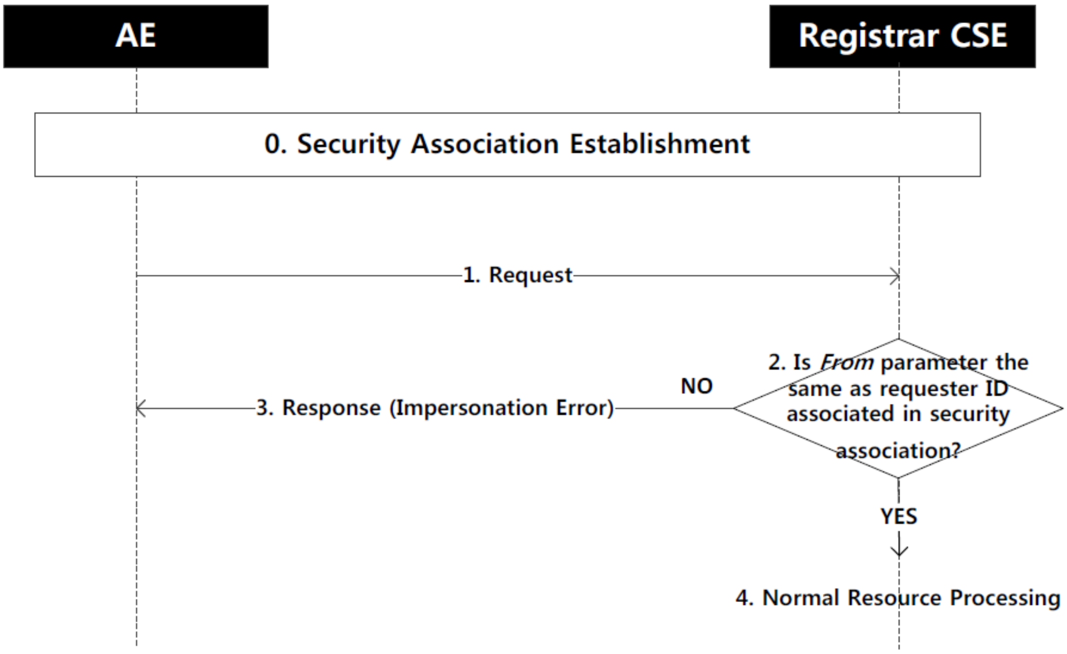
Set of context constraints consisting of the 3 elements:
{accessControlTimeWindow(*k*, *m*), accessControlLocationRegion(*k*,*m*), accessControlIpAddress(*k*, *m*)}

Set of time windows defined by start and end time
Example:
{daily 04:30 – 06:00, 11:30 – 12:30, 22:15 – 00:30}

Set of location regions defined by list of objects representing geographical regions
Example:
{geoRegion1, geoRegion2, geoRegion3}

Set of IP addresses or address blocks
Example (IPv4):
{212.75.201.105, 88.77.0.0/16, 116.27.123.0/24}

# AE Impersonation Prevention



AE

Registrar CSE

0. Security Association Establishment

1. Request

2. Is *From* parameter the same as requester ID associated in security association?

NO

3. Response (Impersonation Error)

YES

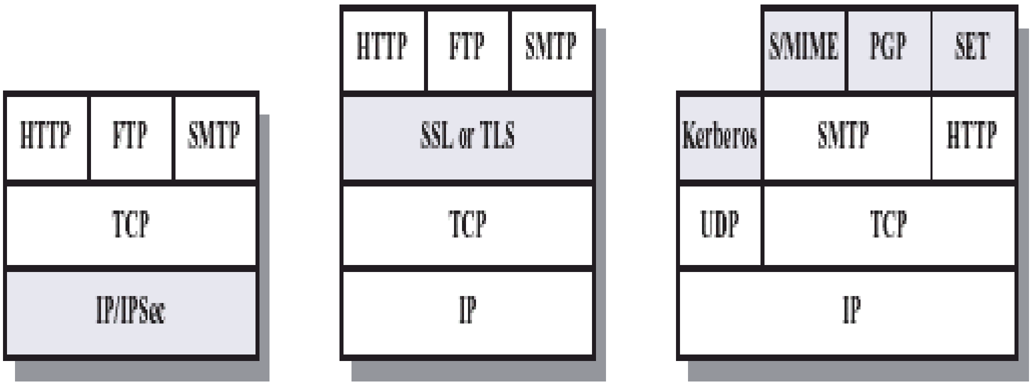
4. Normal Resource Processing

# References

1. D. Boswarthick, O. Elloumi, O. Hersent (Editors),M2M Communications: A Systems Approach, Wiley, 2012
2. 3GPP TS 24.302 - Access to the 3GPP Evolved Packet Core (EPC) via non-3GPP access networks
3. 3GPP TS 33.102 - 3G Security Architecture
4. 3GPP TS 33.402 - System Architecture Evolution (SAE); Security aspects of non-3GPP accesses
5. ETSI TS 102 690 - Machine-to-Machine communications (M2M); Functional Architecture
6. ETSI TS 102 921 - Machine-to-Machine communications (M2M); mIa, dIa and mId interfaces
7. oneM2M TS-0001: Functional Architecture
8. oneM2M TS-0003: Security-Solutions
9. RFC 5246 – The Transport Layer Security (TLS) Protocol Version 1.2
10. RFC 6347 - Datagram Transport Layer Security Version 1.2

# Appendix. TLS/DTLS Security Protocols

# Layers of Security



(a) Network Level

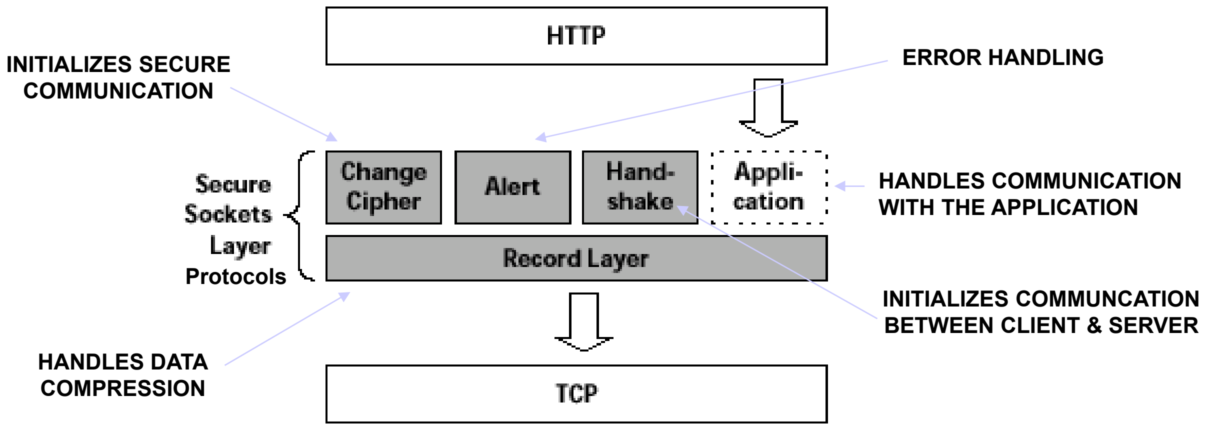
(b) Transport Level

(c) Application Level

# History of TLS

- Evolved from SSL
  - Unreleased v1 (Netscape)
  - Flawed-but-useful v2
  - Version 3 from scratch
  - Standard TLS1.0
    - SSL3.0 with minor tweaks, hence Version field is 3.1
- Defined in RFC2246 (1999), `http://www.ietf.org/rfc/rfc2246.txt`
- Open-source implementation at `http://www.openssl.org/`

# TLS Protocol Architecture



HTTP

INITIALIZES SECURE COMMUNICATION

ERROR HANDLING

Secure Sockets Layer Protocols

Change Cipher

Alert

Hand-shake

Appli-cation

HANDLES COMMUNICATION WITH THE APPLICATION

Record Layer

INITIALIZES COMMUNCATION BETWEEN CLIENT & SERVER

HANDLES DATA COMPRESSION

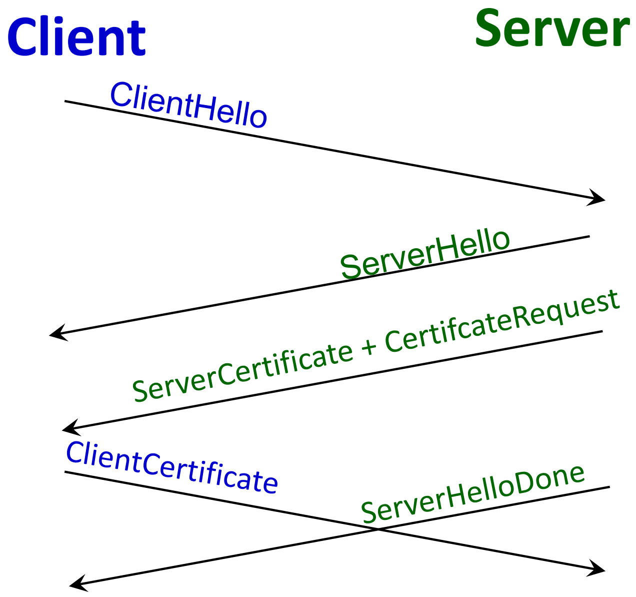
TCP

# TLS Protocol Details

TLS consists of two protocols

- Handshake protocol
  - Use public-key cryptography (or pre-shared key) to establish a shared secret key between the client and the server
- Record protocol
  - Use the secret key established in the handshake protocol to protect communication between the client and the server
- We will focus on the handshake protocol using public-key cryptography.

# TLS Handshake Phase (1)



ClientHello
- List of the ciphersuites* and compression methods
- A random number

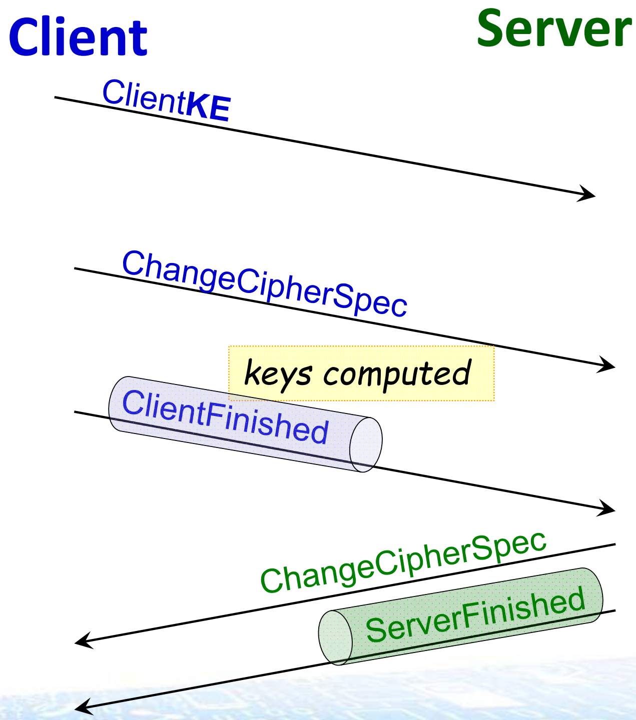
ServerHello
- A choice of the ciphersuite it can support
- A random number (different from client's one)
- Session ID

ServerCertificate
- Name and public key of the server
- Signed by a CA

ClientCertificate
- Name and public key of the server
- Signed by a CA

ServerHelloDone
- Ready to establish a mutual secret key for use in further communications

# TLS Handshake Phase (2)



ClientKE
- Pre-Master Key (Another random number) encrypted by Server’s PK

Client ChangeCipherSpec
- Name and public key of the server
- Signed by a CA

ClientFinished
- Name and public key of the client
- Signed by a CA

Server ChangeCipherSpec

Name and public key of the server

Signed by a CA

ServerFinished

Name and public key of the client

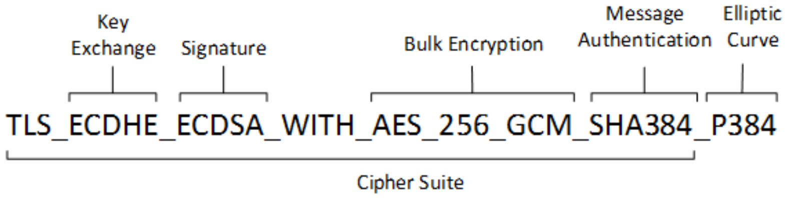
Signed by a CA

# Key Computation

- Generation of Master Key from Clients’ random number, Server’s random number and Pre-Master Key
- Generation of Key Material from Master Key
- Generation of Symmetric Keys and other secret values from Key Material

# *Cipher Suite

- A cipher suite is a combination of a key exchange algorithm, a bulk encryption algorithm, a message authentication code (MAC) algorithm, and a pseudorandom function (PRF).
- The key exchange algorithm, e.g. ECDHE_ECDSA, is used to determine if and how the client and server will authenticate during the handshake.
- The bulk encryption algorithm, e.g. AES_128_GCM, is used to encrypt the message stream. It also includes the key size and the lengths of explicit and implicit initialization vectors (cryptographic nonces).
- The message authentication code algorithm, e.g. SHA256, is used to create the message digest, a cryptographic hash of each block of the message stream.
- The pseudorandom function, e.g. P384, using the MAC algorithm's hash function, is used to create the master secret, a 48-byte secret shared between the two peers in the connection. The master secret is used as a source of entropy when creating session keys, such as the one used to create the MAC.



Key Exchange

Signature

Bulk Encryption

Message Authentication

Elliptic Curve

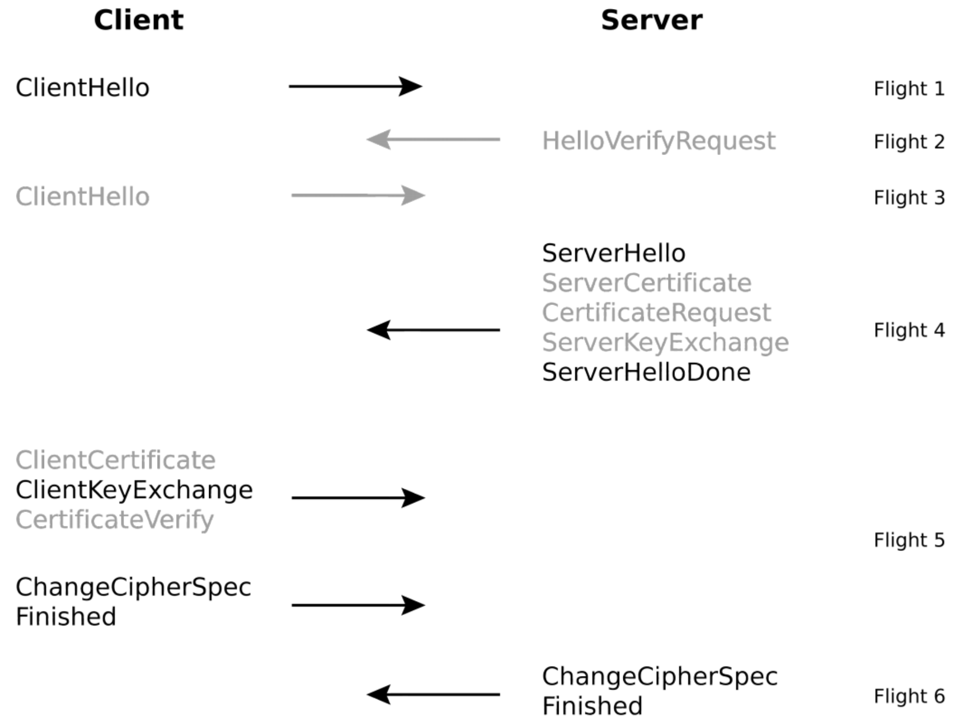
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384_P384

Cipher Suite

# DTLS

- Datagram Transport Layer Security (DTLS) is defined in RFC 4347.
- TLS has been designed for reliable transport protocols, that is it expects no lost or reordered messages from the transport layer.
- If it detects that a message is lost or out of order, it reasonably assumes an attack and drops the connection.
- DTLS is designed to provide reliable transport protocols over unreliable transport protocols where message losses and reordering are very likely.
- It is designed to tolerate unreliability and not creating any security issues while doing so.

# DTLS Handshake Phase



Client

Server

ClientHello

Flight 1

HelloVerifyRequest

Flight 2

ClientHello

Flight 3

ServerHello
ServerCertificate
CertificateRequest
ServerKeyExchange
ServerHelloDone

Flight 4

ClientCertificate
ClientKeyExchange
CertificateVerify

ChangeCipherSpec
Finished

Flight 5

ChangeCipherSpec
Finished

Flight 6

# Considerations of Unreliable Transport – DTLS Flight & Timer

- With an unreliable transport UDP, DTLS has to ensure the reliability of handshake messages itself. Therefore, it needs a timer to retransmit lost messages.
- For increased efficiency, DTLS does not use a timer for every message, but for bundles of messages, called fights.
- A fight contains all messages before the sending side changes. For every fight sent a timer is started, and if there is no response until the timer expires, the entire fight will be retransmitted.

# Other Issues of DTLS

- DTLS uses an additional handshake message, called HelloVerifyRequest to prevent DOS attacks.
- DTLS has to provide its own fragmentation mechanism.
- DTLS has to deal with reordered messages, which can likely occur with unreliable transports.
- DTLS has its own Replay Check.