# Security Common Functions Requirements

Approved Version 1.1 – 31 Jul 2012

**Open Mobile Alliance**

OMA-RD-SEC_CF-V1_1-20120731-A

# Contents

# Figures

# Tables

# 1. Scope (Informative)

This document defines use cases and requirements for OMA Security Common Functions (SEC_CF) 1.1. It describes a set of enhanced or new functional requirements for the enabler Security Common Functions to support OMA enablers.

In order to maintain backward compatibility, this requirement document also contains use cases and requirements in prior release SEC_CF 1.0.

# 2. References

## 2.1 Normative References


**[OMA SEC_CF v1.0]** "OMA Application Layer Security Common Functions V1.0" , Open Mobile Alliance™, OMA-ERP-SEC_CF-V1_0-20080902-A.zip, URL:http://www.openmobilealliance.org/

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


## 2.2 Informative References


**[GBA]** 3GPP TS 33.220 "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Generic Authentication Architecture (GAA); Generic bootstrapping architecture "

URL: http://www.3gpp.org/ftp/Specs/html-info/33220.htm

**[HTTP/1.1]** "Hypertext Transfer Protocol -- HTTP/1.1", IETF RFC 2616, June 1999

URL: http://www.ietf.org/rfc/rfc2616.txt

**[HTTP Digest]** "HTTP Authentication: Basic and Digest Access Authentication", IETF RFC 2617, June 1999

URL: http://www.ietf.org/rfc/rfc2617

**[OMADICT]** "Dictionary for OMA Specifications", Version 2.8, Open Mobile Alliance™, OMA-ORG-Dictionary-V2_8, URL:http://www.openmobilealliance.org/

**[OpenIDAuthentication2.0]** "OpenID Authentication 2.0"

URL: http://openid.net/specs/openid-provider-authentication-policy-extension-1_0.html#OpenIDAuthentication2.0

**[PSK-TLS]** "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)", IETF RFC 4279, December 2005

URL: http://www.ietf.org/rfc/rfc4279

**[TLS]** "Transport Layer Security (TLS) Version 1.0", T. Dierks, E. Rescorla, IETF RFC 2246, Jan 1999, URL:

http://www.ietf.org/rfc/rfc2246.txt

# 3. Terminology and Conventions

## 3.1 Conventions

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

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

## 3.2 Definitions

**Anonymity**

Anonymity provides protection of the identity of a party, against both eavesdroppers and peers

**Identity Protection against Eavesdroppers**:
An attacker (eavesdropper) should not be able to link the communication exchanged by one party to the real identity of the party.

**Identity Protection against Peer**:
The peer in a communication should not be able to link the communication exchanged by one party to the real identity of the party, but rather to an unlinked pseudonym or private identifier.

**Authentication**

**Authentication** is the process of verifying an identity (distinguishing identifier) claimed by or for a system entity, which may be a peer in a communication or the source of some data. This assured Identity may be well known (a real name, telephone number, mailing address, phone number, social security number, IP- or email address) or it can be an unlinkable identifier (like a pseudonym). The verification is achieved presenting authentication information (credentials) that corroborates the binding between the entity and the identifier. Authentication is usually divided into entity and message (or data) authentication. The main difference between the two is that message authentication provides no timeliness guarantee (the authenticated message may be old), while entity authentication implies actual communication with an associated verifier during execution of the current run of the protocol.
Authentication is usually unilateral (“Alice authenticates Bob”). **Mutual Authentication** refers to Authentication in both directions.

**Authorization**

**(by a Trusted Third Party)**

**Authorization** is a right or a permission that is granted to a system entity to access a system resource. An "authorization process" is a procedure for granting such rights.
In some protocols, a Trusted Third Party introduces one principal to another one, and assures to the first one that the second one is trusted and authorized to access the service or function.

**Data Confidentiality**

**Data Confidentiality** is the property that a particular data item or information (usually sent or received as part of the content of a “secured” message, or else constructed on the basis of exchanged data) is not made available or disclosed to unauthorized individuals, entities, or processes, and remains unknown to the intruder. We choose the convention that the secrecy of a session key generated during a key agreement is not considered here but in Goal “Key authentication” above. Also the secrecy of a long-term key used within a protocol is not part considered as a secrecy goal of the protocol.

**Data Integrity**

**Data Integrity** is a security service that protects against unauthorized changes to data, including both intentional change or destruction and accidental change or loss, by ensuring that changes to data are detectable.
A data integrity service can only detect a change and report it to an appropriate system entity; changes cannot be prevented unless the system is perfect (error-free) and no malicious user has access. However, a system that offers data integrity service might also attempt to correct and recover from changes.
Relationship between data integrity service and authentication services: Although data integrity service is defined separately from data origin authentication service and peer entity authentication service, it is closely related to them. Authentication services depend, by definition, on companion data integrity services. Data origin authentication service provides verification that the identity of the original source of a received data unit is as claimed; there can be no such verification if the data unit has been altered. Peer entity authentication service

| | |
|---|---|
| | provides verification that the identity of a peer entity in a current association is as claimed; there can be no such verification if the claimed identity has been altered. |
| **Denial-of-Service (DoS)** | **Denial of Service** attacks target the valuable resources that are needed to provide services. A typical denial of service attack results in the excessive usage of a particular resource by a malicious entity in order to make that resource unusable for the rest of the legitimate users of the service. Below are few examples of DoS attack types:<br><br>DoS on memory allocation,<br><br>DoS on computational power, and<br><br>Overloading attacks on third parties:<br>This is inducing one or several hosts to send large amounts of packets to a victim. |
| **Entity authentication (Peer Entity Authentication)** | **Entity authentication** is assuring one party, through presentation of evidence and/or credentials of the identity of a second party involved in a protocol, and that the second has actually participated during execution of the current run of the protocol. Usually this is done by presenting a piece of data that could only have been generated by the second party in question (as a response to a challenge, for instance). Thus, usually entity authentication implies that some data can be unequivocally traced back to a certain entity, which implies Data Origin Authentication. |
| **Identity Module** | A fixed or removable module keeping identity information and credentials, i.e. a SIM/USIM/ISIM or UIM/RUIM |
| **Key Agreement** | An authenticated key agreement protocol has as goal the secure distribution of keys, and in particular most often session keys. |
| **Message authentication (Data Origin Authentication)** | The protocol must provide means to ensure confidence that a received message or piece of data has been created by a certain party at some (typically unspecified) time in the past, and that this data has not been corrupted or tampered with, but without giving uniqueness or timeliness guarantees. The confidence that data has been created by a certain party, but without the assurance that it has not been modified, is of no interest for us. Thus Message authentication implies integrity. Only very few Internet protocols offer Data Origin Authentication without providing Entity Authentication (IPsec AH or PKI Signatures would be examples). |
| **Privacy** | Privacy is the right of an entity (normally a person), acting in its own behalf, to determine the degree to which it will interact with its environment, including the degree to which the entity is willing to share information about itself with others. (See: anonymity.)<br>In particular, privacy is the right of individuals to control or influence what information related to them may be collected and stored and by whom and to whom that information may be disclosed. |
| **Public-key cryptography** | **Public-key cryptography** (also called asymmetric cryptography) is based on using a pair of two different keys (a public and a private key. A public key is called "public" because it is generally available to everybody and may be used either to encrypt messages intended for the owner of the corresponding private key or to verify the signature of that owner. Corresponding to the public key is a private key, typically known only to one principal. The private key is used to decrypt the message. Because it is uniquely bound to an individual a private key can also be used for a digital signature on a message. But often, for security reasons, different keys and different algorithms are used for decryption and digital signatures. In order to use a public key, the entity using it has to know which principal is bound to the public key. This binding is usually accomplished by a certificate, typically a record asserting such binding, containing an indicator of timeliness and signed by a well-known trusted third party. |
| **Replay Protection** | In a replay attack, the attacker captures one or several messages plays them back to the party which originally received them. The attacker does not need to be able to understand the messages. A protocol provides Replay protection if it offers means to ensure confidence that a received message has not been recorded and played back by an adversary".<br>More precisely, replay protection is assuring one party that an authenticated message is not old. Depending on the context, this could have different meanings:<br><br>that the message was generated during this session, or<br><br>that the message was generated during a known recent time window, or<br><br>that the message has not been accepted before. |

**Symmetric-key cryptography** **Symmetric-key cryptography** (also called secret-key cryptography) relies on the same key for both encryption and decryption.

## 3.3 Abbreviations

| | |
|---|---|
| **CPM** | Converged IP Messaging |
| **DTLS** | Datagram Transport Layer Security |
| **GBA** | Generic Bootstrapping Architecture |
| **MO** | Management Object |
| **OMA** | Open Mobile Alliance |
| **PSK-TLS** | Pre-Shared Key Ciphersuites for Transport Layer Security |
| **SA** | Security Association |
| **SEC_CF** | SECurity Common Functions |
| **SIP** | Session Initiation Protocol |
| **TCP** | Transmission Control Protocol |
| **TLS** | Transport Layer Security |
| **UDP** | User Datagram Protocol |

# 4. Introduction (Informative)

Currently OMA SEC_CF can provide security functionalities (e.g., authentication, confidentiality, data integrity, key management) for OMA Enablers. OMA SEC CF 1.0 [OMA SEC_CF v1.0] works for OMA Enablers that are based on a Client-Server operational model and operate over TCP as the transport protocol. However, some OMA enablers operate over UDP as the transport protocol (e.g., CPM), and some enablers over SIP (e.g., LOCSIP). Therefore, SEC_CF v1.1 will introduce support for UDP, SIP, and Push services in order to support more OMA enablers.

The objective of this document is to collect corresponding use cases to develop a set of enhanced or new functional requirements for the Security Common Functions.

# 5. Security Common Functions release description (Informative)

The Security Common Functions (SEC_CF) Enabler describes a common way to implement security functions for OMA Enablers. These functions shall not be specific to any particular application.

## 5.1 Version 1.0

OMA SEC_CF 1.0 Enabler supports for the following functionality.

- Support for OMA Enablers that are based on a Client-Server operational model
- Support for OMA Enablers over TCP protocol
- GBA Profile
- TLS/PSK-TLS Profile
- SEC-CERT Management Object (MO)

## 5.2 Version 1.1

OMA SEC_CF 1.1 Enabler supports for the following additional functionality

- Support for OMA Push services
- Support for OMA Enablers over SIP protocol
- Support for OMA Enablers over UDP protocol
- Support for Delegated Authentication for Web Services

# 6. Requirements (Normative)

## 6.1 High-Level Functional Requirements

| Label | Description | Release |
|---|---|---|
| SEC_CF-HLF-001 | The SEC_CF enabler SHALL support at least SIP, TCP, UDP, HTTP transports. | SEC_CF 1.1 |
| SEC_CF-HLF-002 | The SEC_CF enabler SHALL support Push and Pull Services. | SEC_CF 1.1 |
| SEC_CF-HLF-003 | The SEC_CF enabler SHOULD support GBA functionality. | SEC_CF 1.1 |

**Table 1: High-Level Functional Requirements**

### 6.1.1 Security

| Label | Description | Release |
|---|---|---|
| SEC_CF-SEC-001 | Any secret data needed to perform the SEC_CF MUST be *stored* such that no unauthorized entity can get access to this data. | SEC_CF 1.0 |
| SEC_CF-SEC-002 | Any secret data needed to perform the SEC_CF MUST be *transmitted* such that no unauthorized entity can access this data. | SEC_CF 1.0 |
| SEC_CF-SEC-003 | It MUST be possible for authorized entities to modify secret data in a secure way. | **SEC_CF 1.0** |
| SEC_CF-SEC-004 | For the push services, SEC_CF SHALL establish a Security Association (SA) from the Server to the Client. | SEC_CF 1.1 |

**Table 2: High-Level Functional Requirements – Security Items**

#### 6.1.1.1 Authentication

| Label | Description | Release |
|---|---|---|
| SEC_CF-AUTH-001 | The SEC_CF MUST be able to provide authentication of the client (requestor) to the responder that makes use of the SEC_CF. Authentication credentials presented by the requestor MUST be communicated to the resource that makes use of the SEC_CF enabler. Mechanisms to communicate these authenticated identities MUST be defined in the SEC_CF specifications. | SEC_CF 1.0 |
| SEC_CF-AUTH-001 a | The SEC_CF MAY be able to provide authentication of the end-user to the resource that makes use of the SEC_CF , e.g. by entering a PIN code, by using biometrics if applicable or a username/password | Future |
| SEC_CF-AUTH-002 | The SEC_CF MUST be able to provide authentication of the resource that makes use of the SEC_CF to the requesting client. Authenticated identities presented by the resource MUST be communicated to the requesting client. Mechanisms to communicate these authentication credentials MUST be defined in the SEC_CF specifications. | SEC_CF 1.0 |
| SEC_CF- AUTH-002 a | It MUST be possible for Authentication (server to client, client to server, or mutual) to be performed via an authentication proxy.. | SEC_CF 1.0 |
| SEC_CF- AUTH-002 b | It MUST be possible for authentication to be performed directly between a client and the resource that makes use of the SEC_CF without an authentication proxy. | SEC_CF 1.1 |

| | | |
|---|---|---|
| SEC_CF- AUTH-002 c | In case that the enabler is distributed between the home network and visited network(s), the SEC_CF MUST be able to provide authentication of the servers (representing the resource) in the visited network to the requesting client. This may be done via a server in the home network assuming a secure connection between the servers is present. | SEC_CF 1.0 |
| SEC_CF- AUTH-003 | The SEC_CF MUST be able to provide data origin authentication. This means, it MUST be possible to ensure confidence that a received message or piece of data has been created by a certain party, and that this data has not been corrupted or tampered with. | SEC_CF 1.0 |
| SEC_CF- AUTH-004 | The SEC_CF MUST be able to provide replay protection to ensure confidence that a received message has not been recorded and played back. | Future |
| SEC_CF- AUTH-005 | The SEC_CF MUST be able to authenticate the source of the broadcast or streaming. | Future |
| SEC_CF- AUTH-006 | The SEC_CF MAY allow the user to authenticate himself to the client, e.g. by entering a PIN code or by using biometrics if applicable. | Future |
| SEC_CF-AUTH-007 | For the push services, SEC_CF SHALL provide authentication of the Server to the Client. | SEC_CF 1.1 |

**Table 3: High-Level Functional Requirements – Authentication Items**

#### 6.1.1.2 Authorization

| Label | Description | Release |
|---|---|---|
| | | |
| | | |

**Table 4: High-Level Functional Requirements – Authorization Items**

#### 6.1.1.3 Data Integrity

| Label | Description | Release |
|---|---|---|
| SEC_CF-INTE-001 | The SEC_CF MUST be able to provide data integrity, i.e. protection against accidental or intentional changes to the data, by ensuring that changes to the data are detectable. The ability of data integrity must be provided for any data transmissions between any resources in either home or visited networks. | SEC_CF 1.0 |
| SEC_CF-INTE-002 | For the push services, SEC_CF SHALL provide data integrity for security parameters pushed from the Server to the Client to establish Security Association (SA). | SEC_CF 1.1 |
| SEC_CF-INTE-003 | For the push services, SEC_CF MAY provide data integrity for user data pushed from the Server to the Client. | SEC_CF 1.1 |

**Table 5: High-Level Functional Requirements – Data Integrity Items**

#### 6.1.1.4 Confidentiality

| Label | Description | Release |
|---|---|---|
| SEC_CF-CONF-001 | The SEC_CF MUST be able to provide data confidentiality that ensures that *transmitted* information is not made available or disclosed to unauthorised individuals, entities, or processes. The ability of data confidentiality must be provided for any data transmissions between any resources in either home or visited networks | SEC_CF 1.0 |
| SEC_CF-CONF-002 | For the push services, SEC_CF SHALL provide data confidentiality for security parameters pushed from the Server to the Client to establish Security Association (SA). | SEC_CF 1.1 |
| SEC_CF-CONF-003 | For the push services, SEC_CF SHALL provide data confidentiality for user data pushed from the Server to the Client. | SEC_CF 1.1 |

**Table 6: High-Level Functional Requirements – Confidentiality Items**

#### 6.1.1.5 Key Management

| Label | Description | Release |
|---|---|---|
| SEC_CF-KEM-001 | The SEC_CF MUST be able to provide a secure means of key agreement prior to key usage. This ability is needed with respect to authentication keys as well as with respect to (temporary) encryption keys and keys needed for data integrity. Affected entities are any resources in either home or visited networks. | SEC_CF 1.0 |
| | | |

**Table 7: High-Level Functional Requirements – Key Management Items**

#### 6.1.1.6 Delegated Authentication for Web Services

| Label | Description | Release |
|---|---|---|
| SEC_CF-DELAUTH-001 | To support OMA enablers using Web Services, the SEC_CF Enabler SHOULD support a delegated authentication model where the application delegates authentication to a trusted party (e.g., using OpenID, OAuth, or SAML). | SEC_CF 1.1 |
| SEC_CF-DELAUTH-002 | The SEC_CF Enabler MAY provide data integrity between the User Agent and the trusted party. | SEC_CF 1.1 |
| SEC_CF-DELAUTH-003 | The SEC_CF Enabler SHOULD provide data confidentiality between the User Agent and the trusted party. | SEC_CF 1.1 |
| SEC_CF-DELAUTH-004 | The SEC_CF Enabler MAY provide pre-shared secret keys for integrity protection between the User Agent and the trusted party. | SEC_CF 1.1 |
| SEC_CF-DELAUTH-005 | The SEC_CF Enabler SHOULD provide pre-shared secret keys for confidentiality protection between the User Agent and the trusted party. | SEC_CF 1.1 |

**Table 8: High-Level Functional Requirements – Delegated Authentication**

### 6.1.2 Charging

| Label | Description | Release |
|---|---|---|
| | | |
| | | |

**Table 9: High-Level Functional Requirements – Charging Items**

### 6.1.3 Administration and Configuration

| Label | Description | Release |
|---|---|---|
| SEC_CF-ADM-001 | It MUST be possible to provide initial keys to the requesters and resources. | SEC_CF 1.0 |
| SEC_CF-ADM-002 | It MUST be possible to change security algorithms in the servers and clients in a secure manner. | SEC_CF 1.0 |

**Table 10: High-Level Functional Requirements – Administration and Configuration Items**

### 6.1.4 Usability

| Label | Description | Release |
|---|---|---|
| SEC_CF-USAB-001 | SEC_CF SHALL provide security in such a way that it introduces minimal complexity, if any, to the End Users of the enabler or service that SEC_CF supports. | SEC_CF 1.1 |
| SEC_CF-USAB-002 | The functionality of SEC_CF SHALL be designed so that it is as easily adoptable, by other OMA enablers, and with as little need for customisation as possible. | SEC_CF 1.1 |

**Table 11: High-Level Functional Requirements – Usability Items**

### 6.1.5 Interoperability

| Label | Description | Release |
|---|---|---|
| | | |
| | | |

**Table 12: High-Level Functional Requirements – Interoperability Items**

### 6.1.6 Privacy

| Label | Description | Release |
|---|---|---|
| | | |
| | | |

**Table 13: High-Level Functional Requirements – Privacy Items**

## 6.2 Overall System Requirements

| Label | Description | Release |
|---|---|---|
| | | |
| | | |

**Table 14: High-Level System Requirements**

# Appendix A. Change History (Informative)

## A.1 Approved Version 1.0 History

| Reference | Date | Description |
|---|---|---|
| OMA-RD-SEC_CF-V1_0-20080902-A | 02 Sep 2008 | Initial document to address the basic starting point<br>Ref TP Doc# OMA-TP-2008-0321-INP_SEC_CF_V1_0_ERP_for_Final_Approval |
| OMA-RD-SEC_CF-V1_1_20120731-A | 31 Jul 2012 | Status changed to Approved by TP<br>Ref TP Doc# OMA-TP-2012-0291-INP_SEC_CF_V1_1_for_Final_Approval |

# Appendix B. Use Cases (Informative)

Editor Note: Two new use cases “Push Enabler Security” and “OpenID for Web Services” are added in Clause B.1 and B.2 respectively. The rest are old use cases copied and pasted from SEC_CF1.0 without any changes (i.e., keeping them as the same format as in the old RD template).

## B.1 Push Enabler Security

The push application, such as push email and mobile advertising, wishes to communicate content to a client (representing a user) of a mobile terminal. The application, acting as a push initiator, will use push enabler to initiate the communication of an event to the user. Such push services are susceptible to all kinds of DoS and replay attacks. So the client has to get help by the Enabler to verify authenticity of the push initiator. Moreover, the client also gets help by the Enabler to decrypt a confidentiality protected (encrypted) push content (e.g., emails).

### B.1.1 Short Description

A user subscribes push service (e.g., push email) from a service provider.

According to the rules made by the user for the push service, the push application will communicate the encrypted content to a client (representing the user). At the same time, some materials required for establishing the secure communication channel might also be pushed to the client.

When receiving the push content, the client will authenticate the push initiator. If the authentication is successful, the client will decrypt the push content. Otherwise, the client will discard this push content.

### B.1.2 Market benefits

User will only receive the push content they subscribed. And the sensitive push content will not be disclosed to others. Therefore, unsolicited message/content will be prevented. DoS and replay attacks will be mitigated.

## B.2 OpenID for Web Services

OpenID is an authentication service for exchanging identity information between endpoints using a digital identifier (e.g., OpenID URL). OpenID eliminates the need for multiple usernames across different websites, simplifying online experience. So, a user can access all websites securely after he/she logins in OpenID service.

OpenID Authentication uses only standard HTTP(S) requests and responses, so it does not require any special capabilities of the User-Agent or other client software.

In order to enable operators to serve as identity providers leveraging their existing infrastructures (e.g., UICC applications and HSS) and to introduce strong authentication to web services, it is necessary to integrate OpenID with the subscriber authentication mechanisms (e.g., IMS AKA, UMTS AKA, EPS AKA, etc.) used in networks.

### B.2.1 Short Description

The following terms are defined in [OpenIDAuthentication2.0]:

Identifier: An Identifier is either a "http" or "https" URI, or an XRI.

User-Agent: The end user's Web browser which implements HTTP/1.1 [RFC2616].

Relying Party (RP): A Web application that wants proof that the end user controls an Identifier, e.g., AOL, Facebook, Paypal, France Telecom, Google, Microsoft, Telecom Italia, etc.

OpenID Provider (OP): An OpenID Authentication server on which a Relying Party relies for an assertion that the end user controls an Identifier, e.g., AOL, Yahoo!, Verisign.

OP Identifier: An Identifier for an OpenID Provider.

Short description of a user accessing web services as below:

The User-Agent delegating a user enters an Identifier in OP (OpenID Provider) when he/she logs onto RP (Relying Party). RP performs discovery based on the identifier provided to identify OP. It then communicates with OP to establish an association and obtain. the shared secret (i.e., the signing key to be used by the OP). RP redirects the User-Agent for authentication. Mutual authentication between the User-Agent and OP runs. After successful authentication, OP redirects the User-Agent back to RP with a signed assertion. RP verifies the assertion and, upon successful verification, grants the User-Agent access to certain resources.

### B.2.2 Market benefits

Integration of OpenID and the subscriber authentication mechanisms used in mobile networks can extend strong authentication to Web Services and allow operators to become identity providers.

## B.3 Shared Key based Security Establishment

### B.3.1 Short Description

A client in a Mobile Terminal establishes a secure connection to an enabler in its home network where the mobile terminal has pre-established credentials. Credentials and their use for the establishment of the secure connection are based on a shared key mechanism such as the Generic Bootstrapping Architecture [GBA]. The connection is either protected with PSK-TLS [PSK-TLS], using a shared key for mutual authentication of the endpoints or TLS 1.0 [TLS] with server certificates for server authentication and a shared key HTTP Digest [HTTP DIGEST] for client authentication.

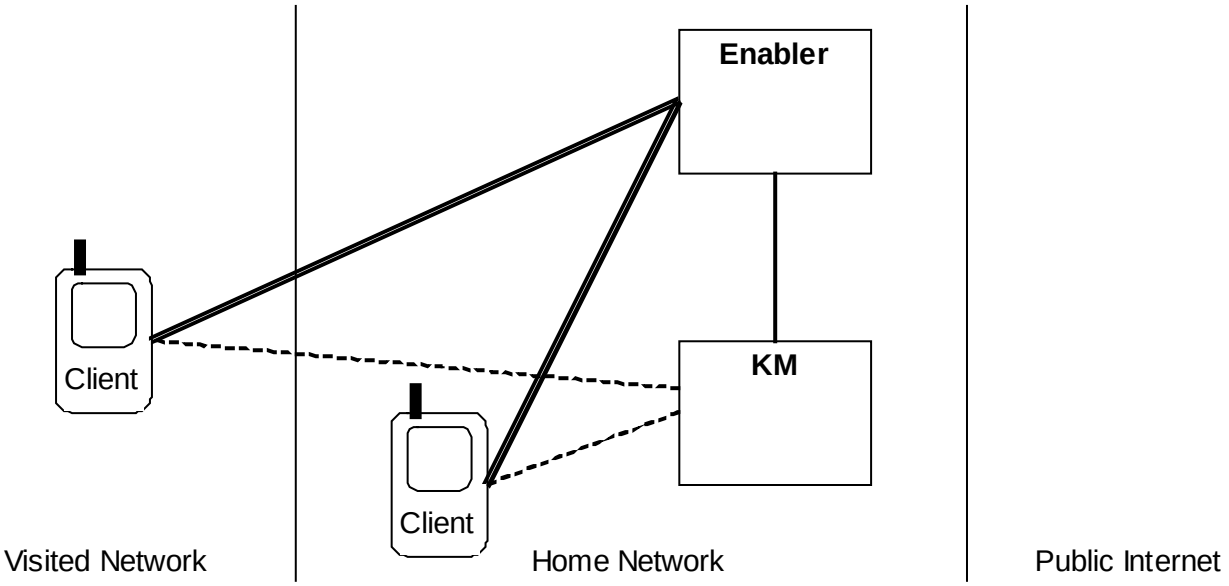


**Figure 1: Secure access to OMA enabler using shared key based key management**

## B.3.2 Actors

We have the following actors in this use case

- The end-user represented by the client in the Mobile Terminal (MT).
- The home network operator. The home network operator runs
  - The enabler function that performs the authentication e.g. in a GBA context is a Network Application Function (NAF)
  - The Key Manager, which performs key generation, management and distribution e.g. in a GBA context is the Bootstrapping Server Function (BSF).
- Possibly a visited network operator. The operator of the visited network is passive and only provides connectivity between the visited and the home network.

### B.3.2.1 Actor Specific Issues

The MT and the home operator have to support a common shared key based mechanism such as GBA functionality.

### B.3.2.2 Actor Specific Benefits

It is essential for users as well as home network operators that users can be offered secure access to services in the their home networks.

## B.3.3 Pre-conditions

The MT can establish TCP connections to the Key Manager and the Enabler. In 3GPP networks, roaming terminals usually have their point of presence in their home networks, which would guarantee that both the Enabler and the Key Manager could be reached from the MT. However, it is sufficient that the MT can directly address the Key Manager and the Enabler and establish a TCP connection. This could always be achieved, even with NAT(P)s in the path, if the Key Manager and the Enabler interfaces had public IP addresses.

Here, it is of course assumed that the Enabler is allowed to use the enabler's key management/authentication functionality. We only note that operators most likely will set up policies governing which enablers that implement the authentication functionality.

## B.3.4 Post-conditions

A TLS protected connection between the MT and the enabler exists. The end points have been mutually authenticated.

## B.3.5 Normal Flow

The MT connects to the Key Manager (e.g.BSF) to retrieve a shared key (e.g. GBA key). The MT then connects to the Enabler and initiates a PSK TLS session, indicating that the key to be used is the retrieved the shared key. The Enabler connects (securely) to the Key Manager and retrieves the indicated shared key together with end-user identity information (anonymous use may be allowed). The shared key is then used in PSK-TLS to establish the payload data protection.

## B.3.6 Alternative Flow

The MT connects to the Key Manager (e.g. BSF) to retrieve a shared key. The MT then connects to the Enabler and initiates a TLS session. The Enabler authenticates itself with a server certificate and requests client authentication with HTTP digest using the agreed/established shared key. The client should validate that the certificate of the Enabler. The Enabler connects (securely) to the Key Manager and retrieves the indicated Enabler (NAF) specific key to be used in the HTTP digest authentication.

### B.3.7 Operational and Quality of Experience Requirements

The establishment of the secure connection should be automatic and invisible to the end-user.

## B.4 Performing Authentication Using an Authentication Proxy

### B.4.1 Short Description

The home operator runs several Enablers that are accessible via HTTP. The home operator uses a common Authentication Proxy (AP) for mutual authentication between enablers and clients. The protected connection between MT and Enabler is terminated in the AP.

A client in a MT establishes a secure connection to the AP in its home network. Credentials and their use for the establishment of the secure connection are based on a shared key management mechanism. The connection is either protected with PSK-TLS [PSK-TLS], using a shared key for mutual authentication of the endpoints or TLS 1.0 with server certificates for server authentication and a shared key HTTP Digest for client authentication.

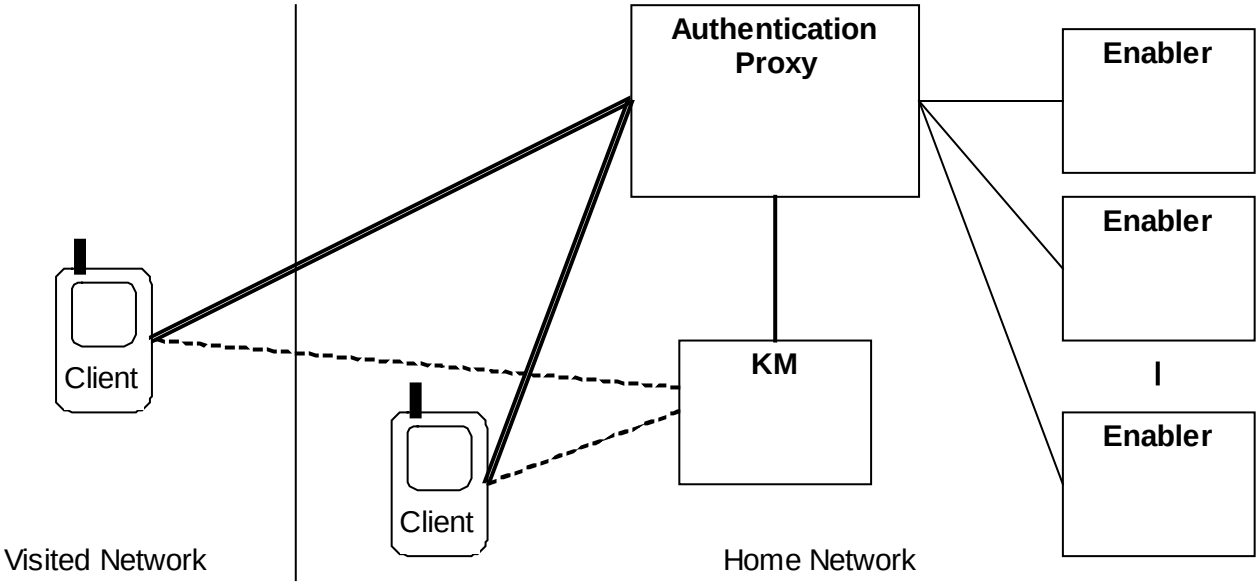


**Figure 2: Access to Enablers via Authentication Proxy**

### B.4.2 Actors

We have the following actors in this use case

- The end-user represented by the client in the MT.
- The home network operator. The home network operator runs
  - The Application Proxy
  - The Key Manager

- Operators of enablers. Usually it is the home network operator that will run the enablers but it is also possible to have 3rd party enablers.
- Possibly a visited network operator. The operator of the visited network is passive and only provides connectivity between the visited and the home network.

### B.4.2.1 Actor Specific Issues

The MT and the home operator have to support a shared key based key management mechanism.

### B.4.2.2 Actor Specific Benefits

It is essential for users as well as home network operators that users can be offered secure access to services in the their home networks. The use of an Authentication Proxy can offload Enablers authentication tasks.

## B.4.3 Pre-conditions

The MT can establish TCP connections to the Key Manager and the AP. In 3GPP networks, roaming terminals usually have their point of presence in their home networks, which would guarantee that both the Enabler and the Key Manager could be reached from the MT. However, it is sufficient that the MT can directly address the Key Manager and the AP and establish a TCP connection. This could always be achieved, even with NAT(P)s in the path, if the Key Manager and the Enabler interfaces had public IP addresses.

Here, it is of course assumed that the AP is allowed to use the shared key management functionality. We only note that operators most likely will set up policies governing which enablers that may be allowed to use the shared key management.

Trusted channels between the AP and the Enablers exist.

## B.4.4 Post-conditions

A TLS protected connection between the MT and the enabler exists. The end points have been mutually authenticated. The Enablers have information about the identity of the end-user, if required.

## B.4.5 Normal Flow

The MT connects to the Key Manager to retrieve a shared key. The MT then connects to the Enabler. This connection is passed via the AP. The MT initiates a PSK-TLS session, indicating that the key to be used is the retrieved the shared key. . The AP connects (securely) to the Key Manager and retrieves the indicated shared key together with end-user identity information (anonymous use may be allowed). The shared key is then used in PSK-TLS to establish payload data protection between the MT and the AP. The AP proxies the traffic from the MT to the intended Enabler together with the user identity information.

## B.4.6 Alternative Flow

The MT connects to the Key Manager to retrieve a shared key. The MT then connects to the Enabler. This connection is passed via the AP. The MT initiates a TLS session. The Enabler authenticates itself with a server certificate and requests client authentication with HTTP Digest with the agreed/established shared key. The client should validate that the server certificate. The AP connects (securely) to the Key Manager and retrieves the indicated Enabler specific key to be used in the HTTP digest authentication. The AP performs user authentication and proxies the traffic from the MT to the intended Enabler together with user identity information.

## B.4.7 Operational and Quality of Experience Requirements

The establishment of the secure connection should be automatic and invisible to the end-user.

# B.5 Certificate based end-user authentication (Optional)

## B.5.1 Short Description

A client in a MT establishes a secure connection to an Enabler. Certificates are used as credentials to establish a TLS connection between the MT and the Enabler.

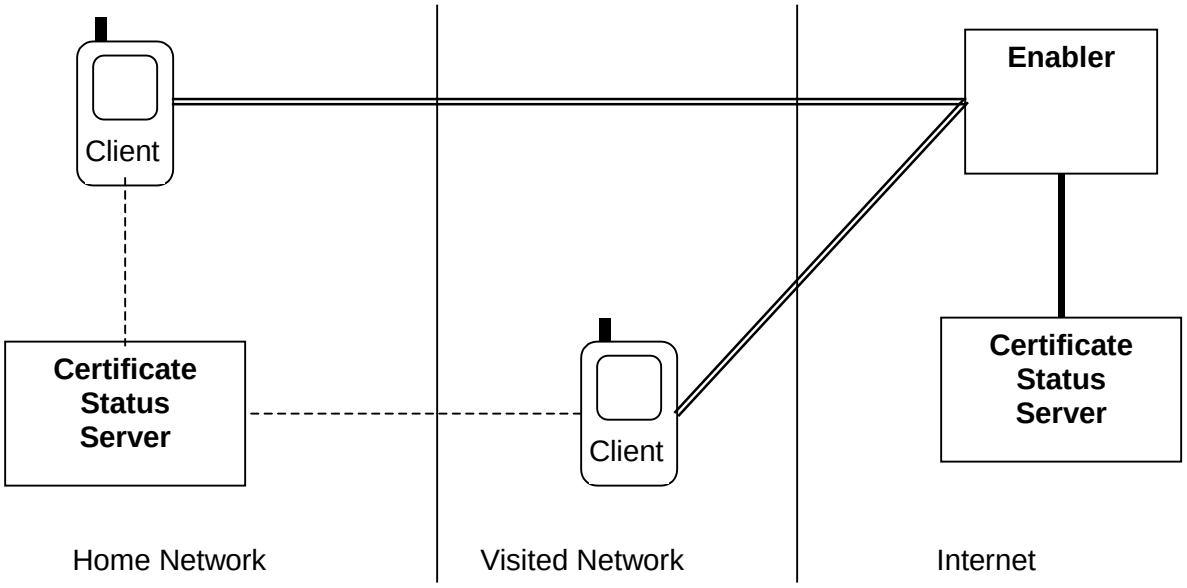


**Figure 3: Use of certificates to establish TLS connection**

## B.5.2 Actors

We have the following actors in this use case

- The end-user represented by the client in the MT.
- The home network operator
- Enabler operator
- Certificate status server operators

### B.5.2.1 Actor Specific Issues

The Certificate Authorities iinvolved issuing client and server certificates have to provide a Certificate Status Service.

### B.5.2.2 Actor Specific Benefits

It is essential for users as well as enabler operators that users can be offered secure access to services.

## B.5.3 Pre-conditions

Client and Enabler certificates are pre provisioned.

The MT has access to a Certifiate Status Server for validation of Enabler certificates. Likewise the Enabler must have access to a Certifiate Status Server for validation of client certificates.

### B.5.4 Post-conditions

A TLS protected connection between the MT and the enabler exists. The end points have been mutually authenticated.

### B.5.5 Normal Flow

The MT connects to the Enabler and triggers a TLS set-up. Client and Enabler (server) certificates are used for mutual authentication.

### B.5.6 Alternative Flow

Void

### B.5.7 Operational and Quality of Experience Requirements

The establishment of the secure connection should be automatic and invisible to the end-user.

## B.6 Distributed Enabler

### B.6.1 Short Description

An enabler is distributed over cooperating parts in different operator domains (one example of such an enabler is Location). The client needs to establish a secure connection to the Enabler function in the visited network. The Enabler function in the home network facilitates the setup of a secure connection between the MT and the Enabler function in the visited network.

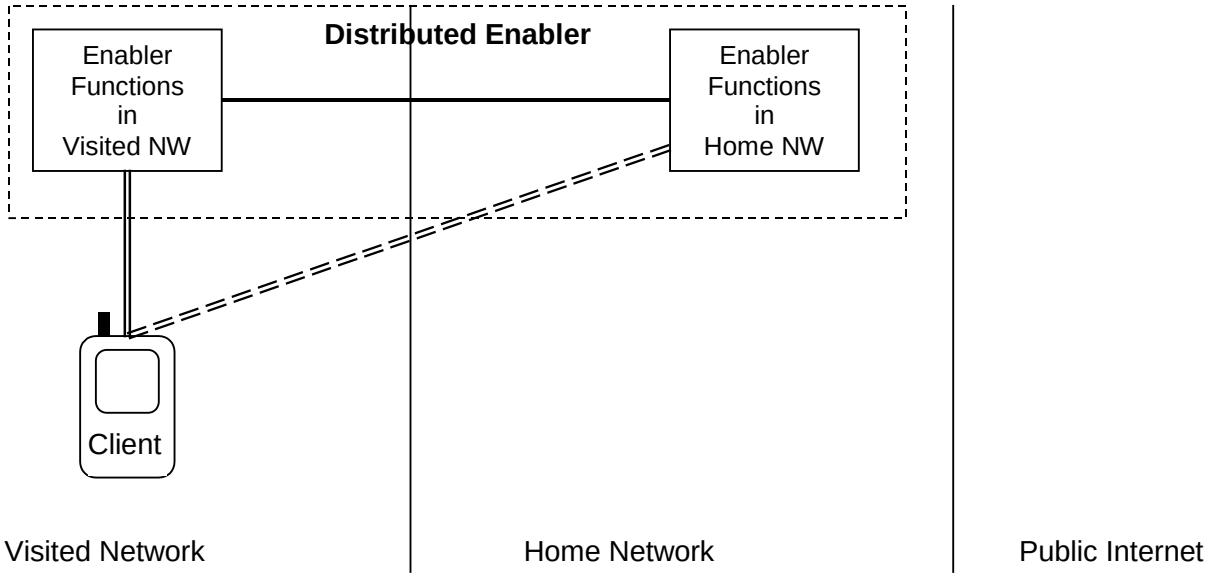


**Figure 4. Distributed enabler accessed in Visited Network**

### B.6.2 Actors

We have the following actors in this use case

- The end-user represented by the client in the MT.

- The home network operator running the Enabler function in the home network.
- The visited network operator running the Enabler function in the visited network.

#### B.6.2.1 Actor Specific Issues

#### B.6.2.2 Actor Specific Benefits

It is essential for users as well as visited network operators that users can be offered secure access to distributed enabler functions located in visited networks.

### B.6.3 Pre-conditions

The Enabler functions in different operator domains have secure channels for inter domain communications. The MT can establish a secure connection to the Enabler function in the home network.

### B.6.4 Post-conditions

A TLS protected connection between the MT and the Enabler function in the the visited network exists. The end points have been mutually authenticated. The Enabler function in the visited network has information about the identity of the end-user, if required.

### B.6.5 Normal Flow

The MT connects securely to the Enbaler function in its home network. How this secure connection is achieved is out of scope in this use case; it could be by use of GBA based key management, preprovisioned secret keys or use of certificates. The Client indicates that it wants to connect to the Enabler function in the visited network. The enabler function in the home network verifies that the visited network enabler function is trusted and generates a key to be used for the setup of a PSK-TLS protected connection between the MT and the Enabler in the visited network. This key and its identity are sent to the MT and to the Enabler in the visited network. The Enabler in the visited network might also obtain information about the end-user identity or other information to authorize its use, if required. Then the MT establishes the PSK-TLS protected connection to the Enabler function in the visited network.

### B.6.6 Alternative Flow

### B.6.7 Operational and Quality of Experience Requirements

The establishment of the secure connection should be automatic and invisible to the end-user.

## B.7 Network initated enabler access.

### B.7.1 Short Description

A service in the network needs access to an enabler function involving the MT. The enabler in the network then initiates that the terminal connects to it by sending a a PUSH message to the MT. The MT connects to the enabler in the home network for verification of the request. Network initiation of services is susceptible to all kind of DoS and replay attacks. Thus the MT has to get help by the Enabler to verify that authencity of the initiation request.

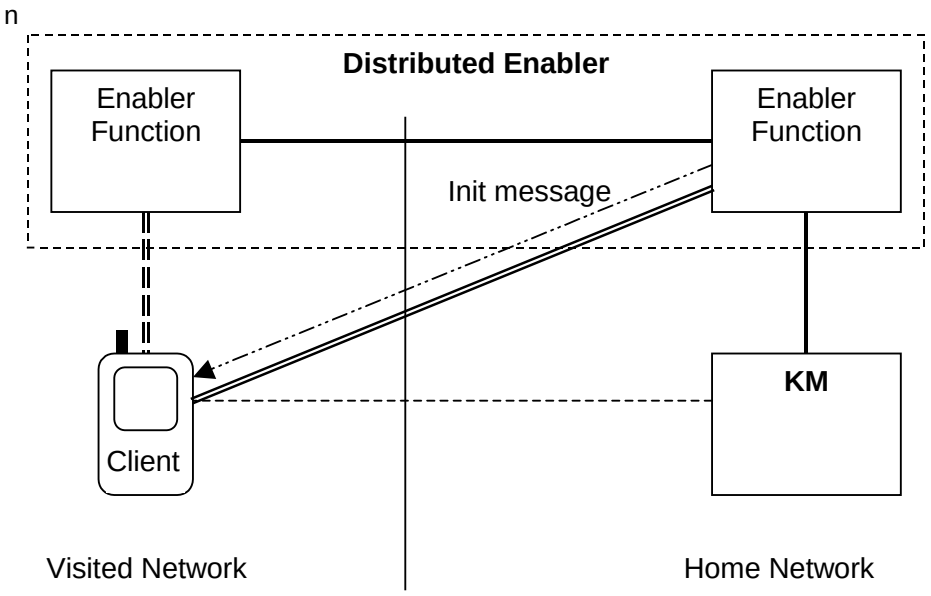


**Figure 5: Network initiated connection between MT and Enabler.**

## B.7.2 Actors

We have the following actors in this use case

- The end-user represented by the client in the MT.
- The home network operator. The home network operator runs
  - The the Enabler or the Enabler function in the home network if the enbaler is distributed over domains
- Possibly a n operator running the visited network part of a distributed Enabler

### B.7.2.1 Actor Specific Issues

### B.7.2.2 Actor Specific Benefits

It is essential for users as well as the operator(s) of an Enabler that there is a way to offer a secure and protected network initiated use of the Enabler.

## B.7.3 Pre-conditions

The MT can establish a secure connection to the Enabler function in the home network. The Enabler functions in different operator domains have secure channels for inter domain communications.

## B.7.4 Post-conditions

The Client in the MT has been assured (by the home network part of the Enabler, that the PUSH message initiating use of the Enabler is valid.

### B.7.5 Normal Flow

The MT receives a PUSH message requesting initiation of a connection to one Enabler function (in the home or the visited network). The MT then connects over a secure channel to the Enabler function in the home network to have the request verified. The Enabler function in the home network indicates that the request is valid (or invalid) and the actual use of the enabler services proceeds.

### B.7.6 Alternative Flow

### B.7.7 Operational and Quality of Experience Requirements

The verification of the validity of the PUSH message initiating the use of the service should be automatic and invisible to the end-user.

## B.8 Provisioning of security parameters.

### B.8.1 Short Description

An Enabler may need to control that a client only can establish connections to or accept connections from trusted entities. Such security controls can be used to prevent the client from being tricked into connecting to fraudulent nodes acting as legitimate enabler entities. The security parameters are usually in the form of white-lists of trusted URL's/URI's for the Enabler, authorized initiators of message exchanges, etc.

This use case is only concerned with the use of device management functionality to achieve the distribution and management of security parameters.

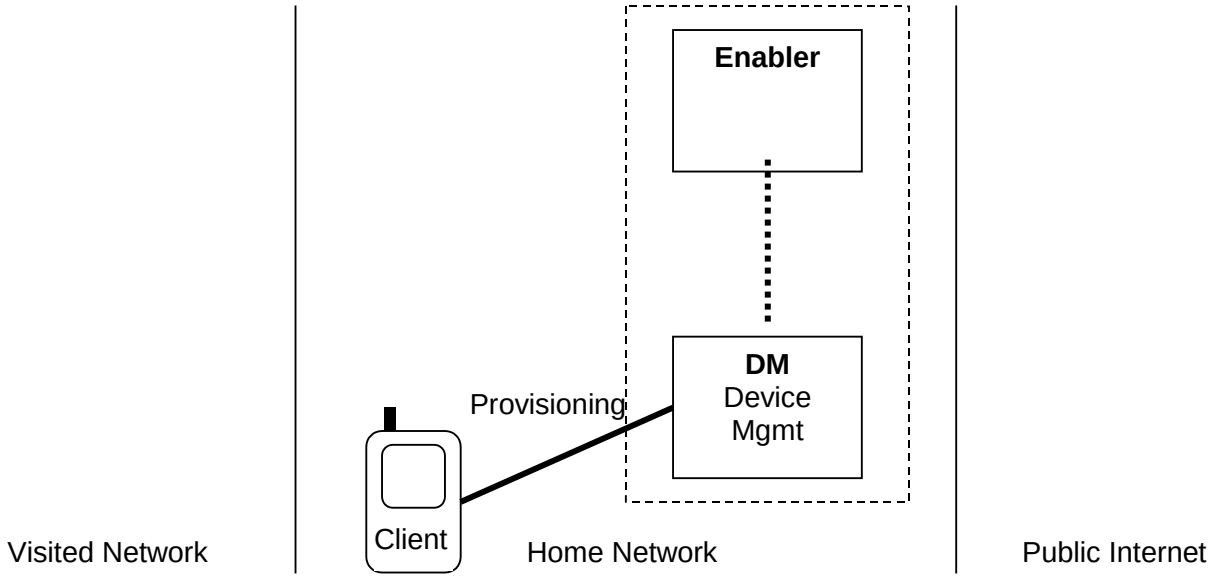


**Figure 6: Provisioning of security parameters.**

### B.8.2 Actors

We have the following actors in this use case

- The end-user represented by the client in the MT.
- The home network operator. The home network operator runs
  - The Enabler
  - The device management system as a stand alone functionality or as part of the Enabler

#### B.8.2.1 Actor Specific Issues

The provisioning of security parameters is based on device management and thus the MT and the Enabler / home network operator has to support a device management system.

#### B.8.2.2 Actor Specific Benefits

It is essential for users as well as the operator(s) of an Enabler that there is a secure way to provision the security parameters required for secure use of the Enabler.

### B.8.3 Pre-conditions

The device management system (MT and Enabler / home network operator) is enabled and configured with adequate security settings. Managed objects for handling of the security parameters are defined.

### B.8.4 Post-conditions

The Enablers managed objects for security parameters in the MT have been populated with data obtained from the Enabler / home network operator.

### B.8.5 Normal Flow

The device management system in the Enabler / home network operator establishes a secure device mangement session with the MT. The device management system writes the Enablers security parameters into the relevant Managed Objects in the MT.

Whenever the Enabler functionality in the MT is invoked, the client in the MT reads the security parameters from the device management system and applies them in its local security control activities.

### B.8.6 Alternative Flow

The device management client in the MT “bootstraps” the security parameters for the Enabler from the Identity Module into its tree of managed objects.

Whenever the Enabler functionality in the MT is invoked, the client in the MT reads the security parameters from the device management system and applies them in its local security control activities.

### B.8.7 Operational and Quality of Experience Requirements

The provisioning of security parameters should be automatic and invisible to the end-user.

## B.9 Provisioning of keys.

### B.9.1 Short Description

An Enabler requires that a secure connection can be established between clients and the enabler. The secret keys needed to establish such a secure connection are provisioned by the Enabler / home network operator. Naming of keys needs to be specified to be compliant with existing key management schemes.
This use case is only concerned with issues of how secret keys can be provisioned to MTs (and clients).

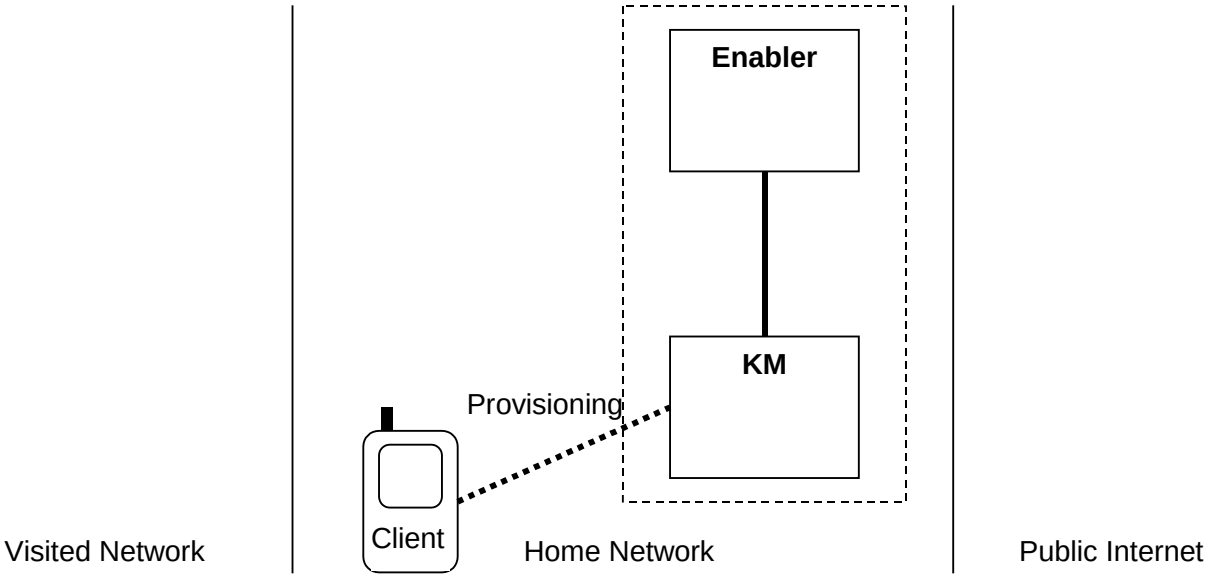


**Figure 7: Provisioning of security parameters.**

## B.9.2 Actors

We have the following actors in this use case

- The end-user represented by the client in the MT.
- The home network operator. The home network operator runs
  - The Enabler
  - The Key Manager

### B.9.2.1 Actor Specific Issues

The provisioning of keys to the MT from the KM may be proprietary and be defined by the home operator, the network or the Enabler. There has to be secure storage for the keys in the MT or in the Identity Module.

### B.9.2.2 Actor Specific Benefits

It is essential for users as well as the operator(s) of an Enabler that there is a secure way to provision the keys required for secure use of the Enabler.

## B.9.3 Pre-conditions

The MT and/or the Identity Module are enabled and configured to receive the keys to be used by the enabler.

## B.9.4 Post-conditions

The MT has securely stored the keys used by the Enabler.

### B.9.5 Normal Flow

The MT/client specific keys used by the Enabler are retrieved from the KM. These keys together with their key identifiers are installed in the MT and/or the IM. When the Enabler service is initiated, the MT uses the installed keys to establish a secure connection to the Enabler. The Enabler requests the corresponding keys from the Key Manager. The set up of the secure connection continues.

### B.9.6 Alternative Flow

### B.9.7 Operational and Quality of Experience Requirements

The provisioning of secret keys security parameters should be automatic and invisible to the end-user.