ETSI TS 133 110 V8.0.0 (2009-01)

Technical Specification

Universal Mobile Telecommunications System (UMTS);
LTE;
Key establishment between a UICC and a terminal
(3GPP TS 33.110 version 8.0.0 Release 8)

Reference
RTS/TSGS-0333110v800

Keywords
LTE, SECURITY, UMTS

ETSI

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

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

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

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

Foreword

This Technical Specification (TS) has been produced by ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities, UMTS identities or GSM identities. These should be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between GSM, UMTS, 3GPP and ETSI identities can be found under http://webapp.etsi.org/key/queryform.asp.

Contents

Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

Introduction

The smart card, tamper resistant device, has a primary role of storing credentials and performing sensitive cryptographic computations, it also provides portability of the user credentials. The smart card is rarely a stand-alone device; it usually interacts with a terminal. Sensitive applications are often split between a smart card and a terminal with sensitive data exchanged between the two. Therefore, the need to establish a secure channel between a UICC and a terminal that may host the UICC or be connected to the device hosting the UICC via a local interface has been identified by different standardization groups in order to protect the communication between the UICC and the terminal.

This document describes key establishment between a UICC and a terminal.

1 Scope

The present document describes the security features and mechanisms to provision a shared key between a UICC and a terminal that may host the UICC or be connected to the device hosting the UICC via a local interface. Candidate applications to use this key establishment mechanism include but are not restricted to secure channel between a UICC and a terminal ETSI TS 102 484 [8].

The scope of this specification includes an architecture overview and the detailed procedure how to establish the shared key between the UICC and the terminal.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[2] 3GPP TS 31.101: "UICC-terminal interface; Physical and logical characteristics".

[3] 3GPP TS 33.220: "Generic Authentication Architecture (GAA); Generic bootstrapping architecture".

[4] 3GPP TS 22.259: "Service requirements for Personal Network Management (PNM); Stage 1".

[5] IETF RFC 2246 (1999): "The TLS Protocol Version 1".

[6] IETF RFC 3546 (2003): "Transport Layer Security (TLS) Extensions".

[7] 3GPP TS 33.222: "Generic Authentication Architecture (GAA); Access to network application functions using Hypertext Transfer Protocol over Transport Layer Security (HTTPS)".

[8] ETSI TS 102 484: "Smart Cards; Secure Channel between a UICC and an end-point Terminal".

[9] 3GPP TS 24.008: "Mobile radio interface Layer 3 specification; Core network protocols; Stage 3".

[10] NIST, FIPS PUB 180-2: "Secure Hash Standard (SHS)".

[11] IETF RFC 4634 (2006): US Secure Hash Algorithms (SHA and HMAC-SHA).

[12] IETF RFC 2104 (1997): "HMAC: Keyed-Hashing for Message Authentication".

[13] 3GPP TR 33.905: "Recommendations for Trusted Open Platforms".

[14] TCG Mobile Phone Specifications, https://www.trustedcomputinggroup.org/specs/mobilephone.

[15] TCG Trusted Network Connect (TNC) Specifications, https://www.trustedcomputinggroup.org/specs/TNC.

[16] 3GPP TS 29.109: "Generic Authentication Architecture (GAA); Zh and Zn Interfaces based on the Diameter protocol; Stage 3".

[17] IETF RFC 2616 (1999): "Hypertext Transfer Protocol -- HTTP/1.1".

[18] IETF RFC 3268 (1999): " Advanced Encryption Standard (AES) Ciphersuites for Transport LayerSecurity (TLS)'

[19] IETF RFC 4279 (2005) "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)".

3 Definitions, symbols and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TR 21.905 [1] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in TR 21.905 [1].

NAF Key Center: Dedicated NAF in charge of performing the key establishment between a UICC and a Terminal.

UICC Hosting Device: The entity, which is physically connected to the UICC. The UICC Hosting Device may be the MT or the ME.

Terminal: For the purposes of the present document, the term Terminal denotes a trusted device that can establish a shared key with a UICC. The Terminal is a generic term aiming to address either the scenario where it is part of the UICC Hosting Device or the scenario where it is a physically separated component (e.g. PNE as defined in TS 22.259 [4]).

Remote Terminal: A Terminal that is physically separated from the UICC Hosting Device.

NOTE: The definition of trusted devices is out of the scope of the specification. It is assumed that the home network can decide whether a terminal is trusted or not.

ICCID: ICCID is the identifier of the smart card. ICCID is defined in ITU standard and is encoded as a 10 octet string.

Terminal_appli_ID: It identifies an application in a Terminal. Terminal_appli_ID is an octet string of maximum 32 octets. If an application has an identifier of longer than 32 octets, this should be hashed using SHA 256 [10] into a string of length 32 octets which will be used as Terminal_appli_ID.

Terminal_ID: It identifies uniquely the Terminal and is 10 octets. The Terminal_ID of a ME is the IMEI and shall be encoded using BCD coding as defined in clause 10.5.1.4 of TS 24.008 [9].

NOTE: In case that the Terminal is not a ME the definition of the type of Terminal_IDs is out of the scope of the specification.

UICC_appli_ID: It uniquely identifies an application in the UICC. The UICC_appli_ID is an octet string of maximum 16 octets.

3.2 Symbols

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

|| Concatenation

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in TR 21.905 [1] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in TR 21.905 [1].

B-TID	Bootstrapping Transaction Identifier
BSF	Bootstrapping Server Function
GBA	Generic Bootstrapping Architecture
GBA_ME	ME-based GBA
GBA_U	GBA with UICC-based enhancements
ICCID	Integrated Circuit Card Identification
KDF	Key Derivation Function
Ks_ext_NAF	Derived key in GBA_U
Ks_int_NAF	Derived key in GBA_U, which remains on UICC
Ks_local	Derived key, which is shared between a Terminal and a UICC
NAF	Network Application Function
MAC	Message Authentication Code
PNE	Personal Network Element
SLF	Subscriber Locator Function
USS	User Security Setting

4 Key Establishment between a UICC and a terminal

4.1 Reference model

GBA_U (TS 33.220 [3]) is used to provision a shared key between a UICC and a Terminal (i.e. Ks_local). The GBA_U key Ks_int_NAF is used by the UICC and the NAF to derive Ks_local. The NAF securely delivers Ks_local to the Terminal through a TLS tunnel, which is established between the NAF and the Terminal.

Figure 4.1 and figure 4.2 show a network model of the entities that utilize the bootstrapped secrets, and the reference points used between them. In figure 4.1 the Terminal is part of the UICC Hosting Device whereas in figure 4.2 the Terminal is connected to the UICC Hosting Device via a local interface.

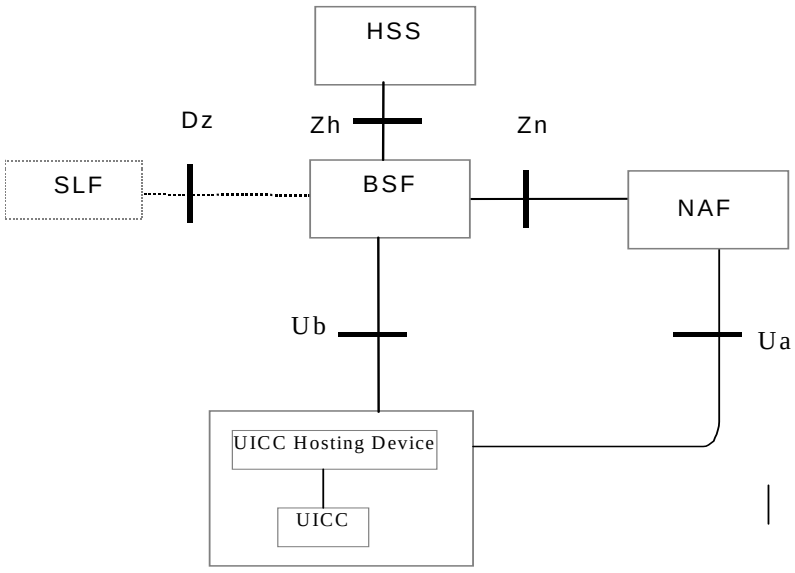

Figure 4.1: High level reference mode (the Terminal is part of the UICC Hosting Device)

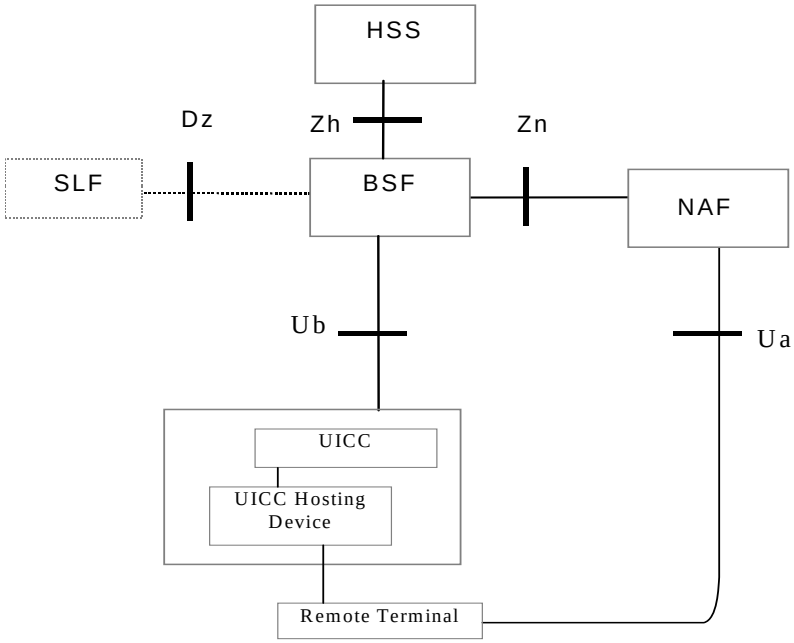

Figure 4.2: High level reference mode (the Remote Terminal is connected to the UICC Hosting Device)

4.2 Network elements

4.2.1 NAF Key Center

The NAF Key Center is the NAF in charge of performing the Key Establishment between a UICC and a Terminal.

4.3 Key establishment architecture and reference points

4.3.1 Reference points

This document is based on the architecture specified in TS 33.220 [3]. The Reference Points that are not explained in this section can be found in TS 33.220 [3] and TS 29.109 [16] (including GAA Service Type Code for this specification).

4.3.2 Reference point Ub

The reference point Ub is implemented between the UICC Hosting Device and the BSF as described in TS 33.220 [3]. The UICC Hosting Device runs the HTTP Digest AKA protocol with BSF. This allows the UICC and the BSF to generate the bootstrapping key Ks.

4.3.3 Reference point Ua

The reference point Ua is used to deliver Ks_local and the associated parameters to the Terminal.

4.4 General requirements and principles for key establishment between a UICC and a Terminal

4.4.1 General requirements

The following requirements and principles are applicable to the procedure for key establishment between a UICC and a Terminal:

- The Terminal and the UICC shall be able to establish a shared key;
- The Terminal shall be trusted;

  NOTE: The definition of trusted terminal is out of scope of the specification. The terminal may be compliant to requirements defined in TCG Mobile Phone specifications [14] or TR 33.905 [13] "Recommendations for Trusted Open Platforms".

- The shared key to establish between the UICC and the Terminal (i.e. Ks_local) shall not be exchanged unencrypted on the interface between the UICC and the Terminal;
- The Terminal and the network shall be able to authenticate each other;
- The server implementing the key establishment function (i.e. the NAF Key Center) needs to be trusted by the home operator to handle the authentication parameters and the shared key;
- The home network shall be able to control whether this Terminal is authorized to establish a shared key with the UICC;
- The procedure for the key establishment between a UICC and a Terminal shall be access independent;
- To the extent possible, existing protocols and infrastructure should be reused;

4.4.2 Requirements on the terminal

The Terminal shall support certificate-based mutual authentication as defined in clause 5.5 of TS 33.222 [7] and IETF RFC 2246 [5] and IETF RFC 3546 [6] in which case the Terminal shall be equipped with a valid Client Certificate or the Terminal shall support shared key based mutual authentication as defined in RFC 4279 [19] in which case the Terminal shall be equipped with a valid pre-shared key.

NOTE: Configuration of certificates and shared secrets is out of scope of the present specification.

4.4.3 Requirements on the UICC hosting device

The UICC Hosting Device shall implement GBA_U as defined in TS 33.220 [3].

4.4.4 Requirements on the UICC

The UICC shall implement GBA_U as defined in TS 33.220 [3].

The UICC shall be capable of deriving Ks_local from Ks_int_NAF.

The NAF_ID of the NAF Key Center shall be stored on the UICC.

NOTE: The home operator can update the NAF_ID of the NAF Key Center by means of OTA commands.

It shall be possible that the UICC implements local policies to restrict the key establishment based on targeted UICC and Terminal applications (i.e. based on Terminal_appli_ID / UICC_appli_ID pair value), or based on Terminal_ID, or based on both targeted applications and Terminal_ID.

4.4.5 Requirements on the NAF Key Center

The NAF Key Center shall support certificate-based mutual authentication as defined in clause 5.5 of TS 33.222 [7] and IETF RFC 2246 [5] and IETF RFC 3546 [6] and shared key based mutual authentication as defined in RFC 4279 [19].

NOTE: Configuration of certificates and shared secrets is out of scope of the present specification.

The NAF Key Center shall be capable of determining whether a Terminal is trusted or not.

The NAF Key Center shall implement GBA_U as defined in TS 33.220 [3].

The NAF Key Center dedicated to the Key Establishment Mechanism shall be located in the operator"s Home Network.

The NAF Key Center shall be capable of deriving Ks_local from Ks_int_NAF.It shall be possible to configure the NAF Key Center to restrict the key establishment based on the targeted UICC and Terminal applications (i.e. based on Terminal_appli_ID / UICC_appli_ID pair value), or based on Terminal_ID and/or ICCID, or based on both targeted applications and device identifiers (Terminal_ID and/or ICCID).

4.4.6 Requirements on Ks_local key and associated parameters handling

The established key Ks_local may be either a key shared between the UICC and the Terminal as monolithic devices or between a specific application on the UICC and a corresponding specific application on the Terminal. Ks_local "per platform" refers to Ks_local shared between the UICC and the Terminal as monolithic devices, whereas Ks_local "per application" refers to Ks_local shared between a specific application on the UICC and a specific application on the Terminal.

Each Ks_local is associated with a Key Lifetime for use in the terminal and a 16 octet Counter Limit value for use in the UICC. The NAF Key Center shall generate these values and deliver them to the terminal. The terminal shall forward the Counter Limit to the UICC when requesting the Ks_local derivation. The Ks_local derivation shall include the Counter Limit value from the NAF Key Center so that the UICC can be sure that the Counter Limit value was generated by the NAF Key Center and was not modified by the terminal. Details of how the UICC shall interpret the Counter Limit can be found in ETSI TS 102 484 [8].

The home operator may update the Ks_local Counter Limit value by means of OTA commands. The description of the OTA mechanism is out of the scope of this TS.

The Terminal shall delete Ks_local and the corresponding parameters (e.g.ICCID, Terminal_appli_ID, UICC_appli_ID) when at least one of the conditions below is met:

1- The key lifetime of Ks_local expires;

2- The Terminal detects that another UICC has been inserted. In order to make this condition possible, the Terminal needs to store in non-volatile memory the last inserted UICC-identity to be able to compare that with the used UICC-identity during the initialisation procedures;

Ks_local should not be deleted from the Terminal when the Terminal is powered down. If the Terminal does not delete Ks_local at power down then Ks_local together with the associated parameters (e.g. key lifetime and B-TID) shall be stored in trusted non-volatile memory.

4.5 Procedures

4.5.1 Initiation of key establishment between a UICC and a Terminal

Before a Ks_local-based application can start, the UICC and the Terminal first have to share the same key Ks_local associated to the selected application. The Terminal shall check if it stores the key Ks_local associated to the targeted application and if this key Ks_local is also available on the UICC.

1- The Terminal checks if it already stores a valid key Ks_local required for the application communicating with the UICC. If a valid key Ks_local is not available on the Terminal then the Terminal initiates a Key Establishment procedure, else step 2 applies.

2- The Terminal sends a request to the UICC to check that the required key Ks_local is available on the UICC. The UICC reply indicates the Terminal if the required key Ks_local is available on the UICC. If the required key Ks_local is not available on the UICC, the Terminal initiates a key establishment procedure, else a valid Ks_local key is shared between the UICC and the Terminal.

4.5.2 Key establishment procedure

If a key establishment procedure is needed, it has to be performed as follows:

1- The Terminal checks whether there is a valid Ks key in the UICC, by fetching the current B-TID and its corresponding lifetime from the UICC. If no valid key Ks is available in the UICC, the Terminal requests a GBA bootstrapping procedure run to derive a new Ks key in the UICC and the BSF.

2- In order to check whether there is a valid Ks_int_NAF, the Terminal sends a request to the UICC to retrieve B-TID value associated to the NAF_ID of the NAF Key Center. In case that the Terminal does not know the NAF_ID of the NAF Key Center, the Terminal sends a request to the UICC to retrieve the NAF_ID of the NAF Key Center.

3- The UICC returns the NAF_ID and associated B-TID to the Terminal. If there is no Ks_int_NAF available in the UICC, a GBA_U NAF Derivation procedure associated to the NAF Key Center is performed and then the UICC returns the NAF_ID and associated B-TID to the Terminal.

4- The Terminal and the NAF Key Center establish a secure tunnel. The secure tunnel may be a HTTPS tunnel with certificate based mutual authentication between the Terminal and NAF Key Center as defined in clause 5.5 of TS 33.222 [7], or based on a shared key based mutual authentication between the Terminal and the NAF Key Center as defined in RFC 4279 [19].

NOTE 1: One potential way to reach a trusted state is if the Terminal is compliant with the requirements defined in TCG (Trusted Computing Group) MPWG (Mobile Phone Working Group) Mobile Phone Specifications [14]. In PC-based TCG technology [15], HTTPS tunnel establishment can be bound to the trust status of the Terminal, through the attestations of relevant trusted engine of the Terminal. Thus, HTTPS tunnel establishment may in future be possible only if the Terminal is in a trusted state.

The psk_identity_hint shall be used by the server to indicate to the PSK TLS client which PSK to use. The pre-shared key is pre-administrated to the Terminal and NAF Key Center.

NOTE 2: If other PSKs are allowed, then the psk_identity_hint needs to be specified in the relevant key specifications.

If several PSKs are allowed, then the different hints are separated by semi-colon. The usage of the psk_identity_hint in PSK TLS handshake is out of scope of this specification.

5- In order to retrieve Ks_local from the NAF Key Center, the Terminal sends a "service request" message to the NAF Key Center node in the mobile operator network. The message is sent within HTTPS tunnel.

The request shall contain the following payload: the identity (B-TID), the Terminal identifier (Terminal_ID), the smart card identifier (ICCID), the application identifier of UICC application (UICC_appli_ID) and the application identifier of the Terminal application (Terminal_appli_ID) requiring the establishment of key Ks_local, and a variable value RANDx.

In case that Ks_local has to be established per platform, the UICC_appli_ID and the Terminal_appli_ID octet strings equal to static ASCII-encoded string "platform".

NOTE 2: The variable value RANDx can be a random value or timestamp produced by the Terminal.

6- The NAF Key Center shall behave as follows:

a) If the Terminal_ID is IMEI, then the NAF Key Center shall check if the Terminal_ID is blocked (blacklisted) and if so, it shall not proceed with the key establishment procedure but respond with an appropriate error code and terminate the secure connection with the Terminal. If the Terminal_ID is not IMEI, then the NAF

Key Center should, if applicable, check if the Terminal_ID is blocked (blacklisted) and if so, it shall not proceed with the key establishment procedure but respond with an appropriate error code and terminate the secure connection with the Terminal. If the key establishment procedure is not allowed for the targeted applications , then the NAF Key Center shall not proceed with the key establishment procedure but respond with appropriate error code and terminate the TLS connection with the Terminal.

NOTE 3: Details of how blacklisting is implemented are out of scope of the present specification.

b) The NAF Key Center contacts the BSF and sends the identity (B-TID) and its own NAF_ID in a credential request.

7- The BSF derives Ks_int_NAF, Ks_ext_NAF and supplies to the NAF Key Center the requested keys Ks_int/ext_NAF keys, as well as the bootstrapping time and the key lifetime of Ks_int/ext_NAF keys.

The BSF may also send requested USSs to NAF Key Center according to the BSF"s policy

8- The NAF Key Center shall behave as follows

a) If the NAF Key Center has requested a USS, and the USS indicates to the NAF Key Center that the key establishment procedure is not allowed for the user, then the NAF Key Center shall respond with appropriate error code and terminate the TLS connection with the Terminal.

b) The NAF Key Center generates a suitable 16 octet Counter Limit for use in the UICC. The NAF Key Center associates a key lifetime to the derived key Ks_local for use in the Terminal.

c) The NAF Key Center derives Ks_local from Ks_int_NAF. Ks_local is computed as Ks_local = KDF (Ks_int_NAF, B-TID, Terminal_ID, ICCID, Terminal_appli_ID, UICC_appli_ID, RANDx, Counter Limit), where KDF is the key derivation function as specified in Annex A.

NOTE 4: If two applications on the UICC or on the Terminal have the same application identifier and RANDx is not renewed for each Ks_local derivation, then Ks_local will be the same for the two applications.

9- The NAF Key Center sends within HTTPS tunnel a response message to the Terminal with the following payload: B-TID, Ks_local, Key Lifetime, and Counter Limit.

10- The Terminal stores Ks_local and associated parameters Key Lifetime, ICCID, Terminal_appli_ID, UICC_appli_ID

11- The Terminal sends a command to perform Ks_local derivation on the UICC. The Terminal sends the NAF_ID corresponding to the NAF Key Center, the Terminal_ID, the Terminal_appli_ID, the UICC_appli_ID, RANDx and the Counter Limit value. The terminal also includes a MAC which is computed as MAC = HMAC-SHA-256(Ks_local, NAF_ID || Terminal_ID || ICCID || Term_appli_ID || UICC_appli_ID || RANDx || Counter Limit) truncated to 16 octets, where HMAC-SHA-256 with truncation is defined in NIST, FIPS PUB 180-2 [10], IETF RFC 4634 [11] and IETF RFC 2104 [12].

Terminal_appli_ID and UICC_appli_ID correspond to identifiers of applications that aim at sharing a key Ks_local. In case that Ks_local has to be established per platform, the UICC_appli_ID and the Terminal_appli_ID octet strings are set equal to the static ASCII-encoded string "platform".

12- The UICC retrieves the Ks_int_NAF and B-TID associated with the received NAF_ID. The UICC may store a local policy to determine the associations between a Terminal_appli_ID and a UICC_appli_ID which are authorized. If the Terminal requested a Terminal_appli_ID/UICC_appli_ID association not authorized by the UICC policy, then the UICC stops the key establishment procedure and returns a "not authorized" error message. The local policy may also not authorize the key establishment procedure based on the Terminal_ID value.

If the requested association is authorised, then the UICC derives Ks_local. Ks_local is computed in the UICC as Ks_local = KDF (Ks_int_NAF, B-TID, Terminal_ID, ICCID, Terminal_appli_ID, UICC_appli_ID, RANDx, Counter Limit), where KDF is the key derivation function specified in Annex A.

The UICC verifies the MAC value received from the Terminal by computing MAC' = HMAC-SHA-256(Ks_local, NAF_ID || Terminal_ID || ICCID || Term_appli_ID || UICC_appli_ID || RANDx || Counter Limit) truncated to 16 octets. If the MAC' does not equal MAC, then the UICC terminates the key agreement procedure and returns a MAC verification failure message in response to the Ks_local derivation request.

13- If MAC'=MAC then the UICC stores Ks_local and associated parameters Terminal_ID, Terminal_appli_ID, UICC_appli_ID and the Ks_local Counter Limit. The UICC then sends a Ks_local derivation response containing a MAC of the ASCII-encoded string "verification successful" using the key Ks_local and the MAC algorithm HMAC-SHA-256 [11] truncated to 16 octets IETF RFC 2104 [12].

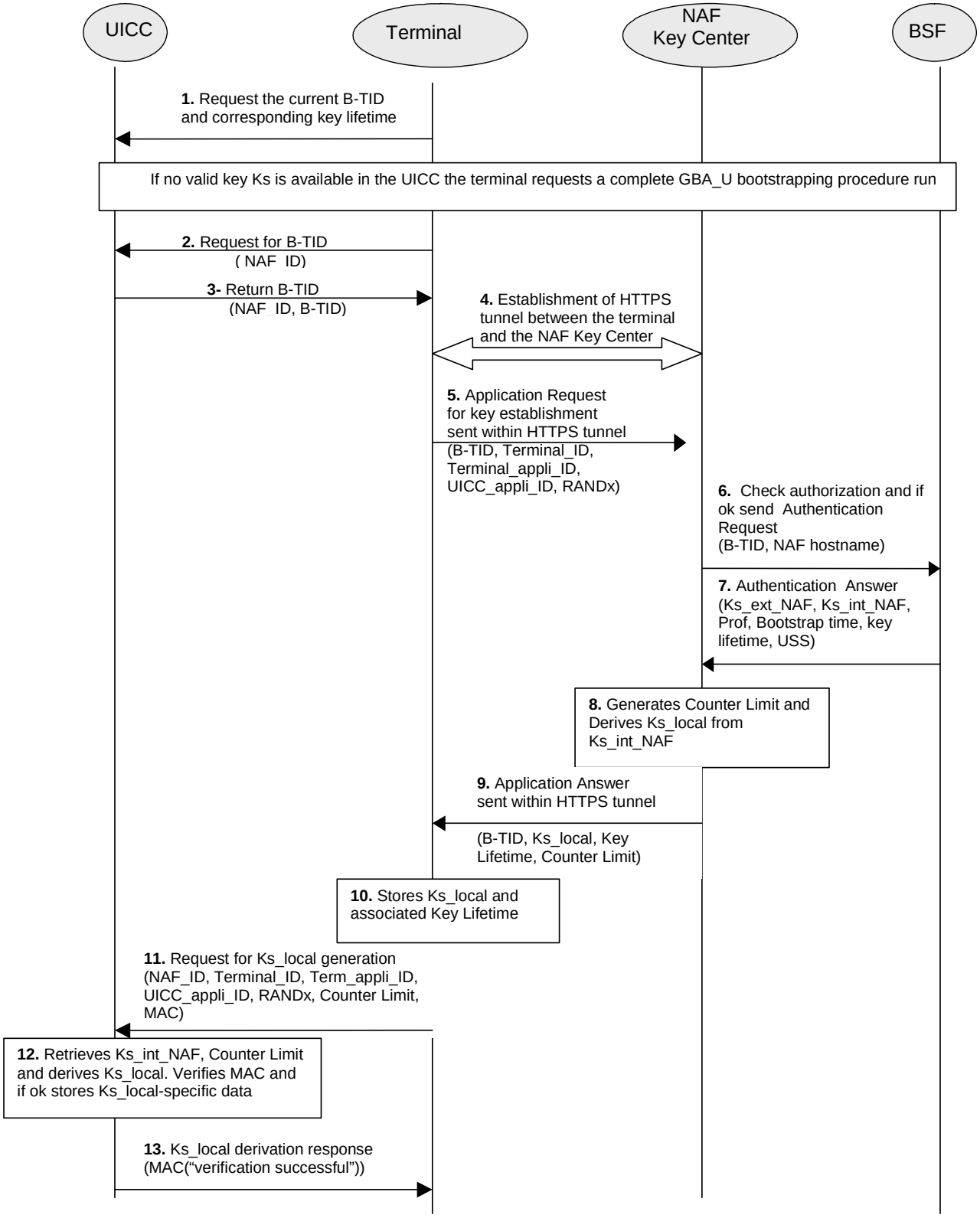

Figure 4-3: Key establishment procedure

Annex A (normative): Key Derivation Function definition

A.1 Ks_local key derivation in key establishment

The description of key derivation function KDF can be found in TS 33.220 [3]. The generic key derivation function and input parameter encoding in this document shall be implemented as defined in TS 33.220 [3].

A.2 Input parameters for Ks_local key derivation

In the key establishment between a UICC and a terminal, the input parameters for the key derivation function shall be the following:

- FC = 0x01,
- P0 = B-TID,
- L0 = length of B-TID is variable (not greater than 65535),
- P1 = Terminal_ID,
- L1 = length of Terminal ID is variable (not greater than 10 octets),
- P2 = ICCID,
- L2 = length of ICCID is variable (not greater than 10 octets),
- P3 = Terminal_appli_ID,
- L3 = length of Terminal_appli_ID is variable (not greater than 32 octets),
- P4 = UICC_appli_ID,
- L4 = length of UICC_appli_ID is variable (not greater than 16 octets),
- P5 = RANDx,
- L5 = length of RANDx is variable (not greater than 16 octets).
- P6 = Counter Limit.
- L6 = length of Counter Limit is 16 octets.

In case that derived key Ks_local has to be established per platform, the UICC_appli_ID and the Terminal_appli_ID octet strings equal to static ASCII-encoded string "platform".

Annex B (normative): Key establishment UICC-Terminal interface

This annex describes the UICC-Terminal interface to be used to derive Ks_local key in the UICC when there is the establishement of a shared key Ks_local between a UICC and a Terminal.

B.1 Local Key Establishment: Key Derivation procedure

This procedure is part of the key establishment to share Ks_local key between a UICC and a Terminal.

The Terminal has previously performed a GBA_U bootstrapping procedure and subsequent GBA_U NAF Derivation procedure, as described in TS 33.220 [3], with the NAF Key Center. The UICC stores the corresponding Ks_int_NAF and associated B-TID together with the NAF_ID of the NAF Key Center.

The NAF_ID of the NAF Key Center is stored on the UICC. This value shall be accessible by the Terminal.

The Terminal sends to the UICC the list of parameters described in the Terminal request for Ks_local generation in clause 4.5.2.

The UICC uses the NAF_ID to retrieve Ks_int_NAF associated to the NAF Key Center. The UICC derives Ks_local from Ks_int_NAF as described in clause 4.5.2.

After successful Ks_local key derivation, the UICC stores Ks_local and associated parameters as described in clause 4.5.2.

Ks_local identifier consists of the concatenation of NAF_ID, Terminal_ID, UICC_ID, Term_appli_ID, UICC_appli_ID and RANDx.

In case that the UICC does not have enough storage available for the generated Ks_local and associated parameters, the UICC shall overwrite an existing Ks_local and associated parameters. To determine the Ks_local to overwrite, the UICC shall construct a list of Ks_local identifiers by storing in the list first position the Ks_local identifier of the last used or derived Ks_local and by shifting down the remaining list elements. The last Ks_local identifier in this list corresponds to the Ks_local to overwrite when the UICC runs out of free memofy. If an existing Ks_local in use is overwritten, the application using Ks_local shall not be affected.

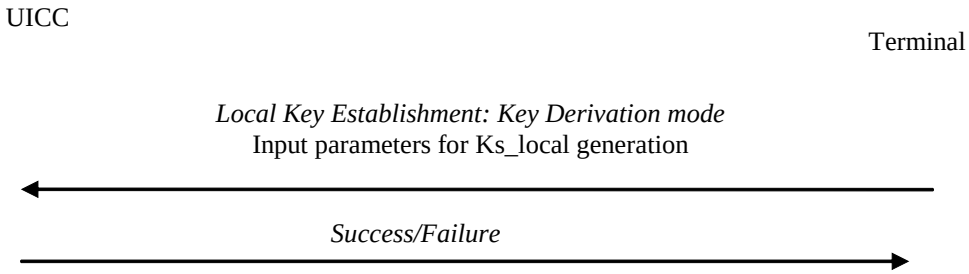

Figure B.1

B.2 Local Key Establishment: Key Availability Check procedure

This procedure takes place during the initiation of the key establishment procedure where the Terminal checks if the UICC already stores a valid key Ks_local required for the application communicating with the UICC.

The UICC has previously performed a Key Derivation procedure for the local key establishment.

The Terminal sends either a Key Identifier of Ks_local or no parameter.

The UICC checks if a corresponding valid Ks_local is available. If a valid Ks_local key is available a "key availability status operation" is returned. In case no valid Ks_local key is available the command fails and a failure message is returned.

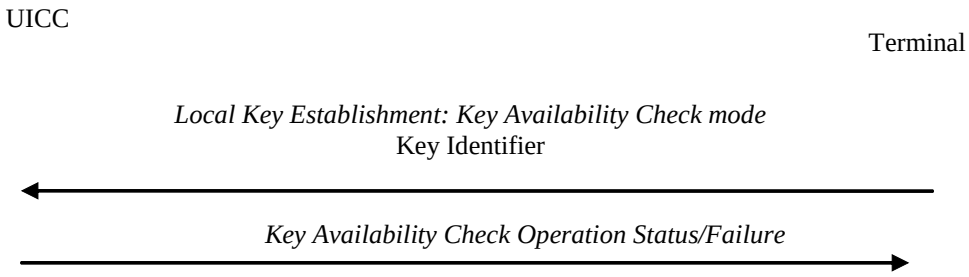

Figure B.2

Annex C (normative): HTTP based key request procedure

C.1 Introduction

Clause 4.5 specifies the HTTP based Key request procedure between the NAF Key Center and the Terminal. It specifies how the Terminal retrieves the Ks_local key from the NAF Key Center together with some associated parameters.

C.2 Key request procedure

This clause contains the following HTTP based procedures:

- Key request;

C.2.1 Key request

The Terminal shall generate a request for Key Request according to clause 4.5.2. The Terminal shall send the Key Request to the NAF Key Center in the HTTP payload in a HTTP POST request. The Request-URI shall indicate the type of the message, i.e. Key Request. Upon successful request, NAF Key Center shall return indication of success together with the Ks_local key.

The Terminal populates the HTTP POST request as follows:

- the HTTP version shall be 1.1 which is specified in RFC 2616 [17];
- the base of the Request-URI shall contain the full NAF Key Center key establishmentURI (e.g. http://nafkeyCenter.home1.net:1234);
- the Request-URI shall contain an URI parameter "requesttype" that shall be set to "key-request-UICCkey", i.e. Request-URI takes the form of "/keyestablishment?requesttype=key-request-UICCkey";
- the Terminal may add additional URI parameters to the Request-URI;
- the HTTP header Content-Type shall be the MIME type of the payload, i.e. "application/keyest-UICCkeyrequest+xml". The XML schema of the payload is specified in Annex E.1 in this specification;
- the Terminal may add additional HTTP headers to the HTTP POST request.

The Terminal sends the HTTP POST to the NAF Key Center. The NAF Key Center checks that the HTTP POST is valid, and extracts the Key request for further processing. The NAF Key Center shall verify that the Terminal is authorized to use this service according to clause 4.5.2.

Upon successful authorization verification, the NAF Key Center shall return the HTTP 200 OK to the Terminal.

The NAF Key Center shall populate HTTP response as follows:

- the HTTP status code in the HTTP status line shall be 200;
- the HTTP header Content-Type shall be the MIME type of the payload, i.e. "application/keyest-keyresponse+xml". The XML schema of the payload is specified in Annex E.2 in this specification;

The NAF Key Center shall send the HTTP response to the Terminal. The Terminal shall check that the HTTP response is valid.

C.2.2 Error situations

The key request procedure may not be successful for multiple reasons. The error cases are indicated by using 4xx and 5xx HTTP Status Codes as defined in RFC 2616 [17]. The 4xx status code indicates that the Terminal seems to have erred, and the 5xx status code indicates that the NAF Key Center is aware that it has erred. Possible error situations during key establishment and their mappings to HTTP Status Codes are described in table C.2.2.1.

NOTE: In table C.2.2.1, the "Description" column describes the error situation in NAF Key Center. The "NAF Key Center error" column describes the typical reason for the error.

The NAF Key Center shall send the HTTP response to the Terminal. The Terminal shall check that the HTTP response is valid.

Table C.2.2-1: HTTP Status Codes used for key request errors

HTTP Status Code	HTTP Error	Terminal should repeat the request	Description	NAF Key Center error
400	Bad Request	No	Request could not be understood	Request was missing, or malformed
401	Unauthorized	Yes	Not used by NAF Key Center	-
402	Payment Required	No	Not used by NAF Key Center	-
403	Forbidden	No	NAF Key Center understood the request, but is refusing to fulfil it	The request was valid, but Terminal is not allowed to use this service
404	Not Found	No	NAF Key Center has not found anything matching the Request-URI	The Request-URI was malformed and NAF Key Center cannot fulfil the request
405	Method not allowed	No	The method specified in the Request-Line is not allowed for the resource identified by the Request-URI.	
406 to 417	*	No	Not used by NAF Key Center	-
500	Internal Server Error	No	Not used by NAF Key Center	-
501	Not Implemented	No	NAF Key Center does not support the requested functionality	The server does not contain particular NAF Key Center service requested
502	Bad Gateway	No	Not used by NAF Key Center	-
503	Service Unavailable	Yes	NAF Key Center service is currently unavailable	NAF Key Center is temporarily unavailable, Terminal may repeat the request after delay indicated by "Retry-After" header
504	Gateway Timeout	No	The server, while acting as a gateway or proxy, did not receive a timely response from the upstream server	The NAF Key Center did not get response over Zn interface.
505	HTTP Version Not Supported	No	NAF Key Center does not support the HTTP protocol version that was used in the request line	Terminal should use HTTP/1.1 version with NAF Key Center

Annex D (informative): Signalling flows for key request procedure

D.1 Introduction

This annex gives examples of signalling flows for using HTTP. It is assumed that TLS with certificate based mutual-authentication has been established before the HTTP signalling flow described in this section takes place.

The Terminal requests a Ks_local key from the NAF Key Center and an example of the signalling flow of the key request procedure between the Terminal and the NAF Key Center is given in clause D.2.

D.2 Signalling flow demonstrating a successful key request procedure

The signalling flow in figure D.2-1 describes the generic message exchange between a Terminal and an NAF Key Center using HTTP. The HTTP client application resides in the Terminal.

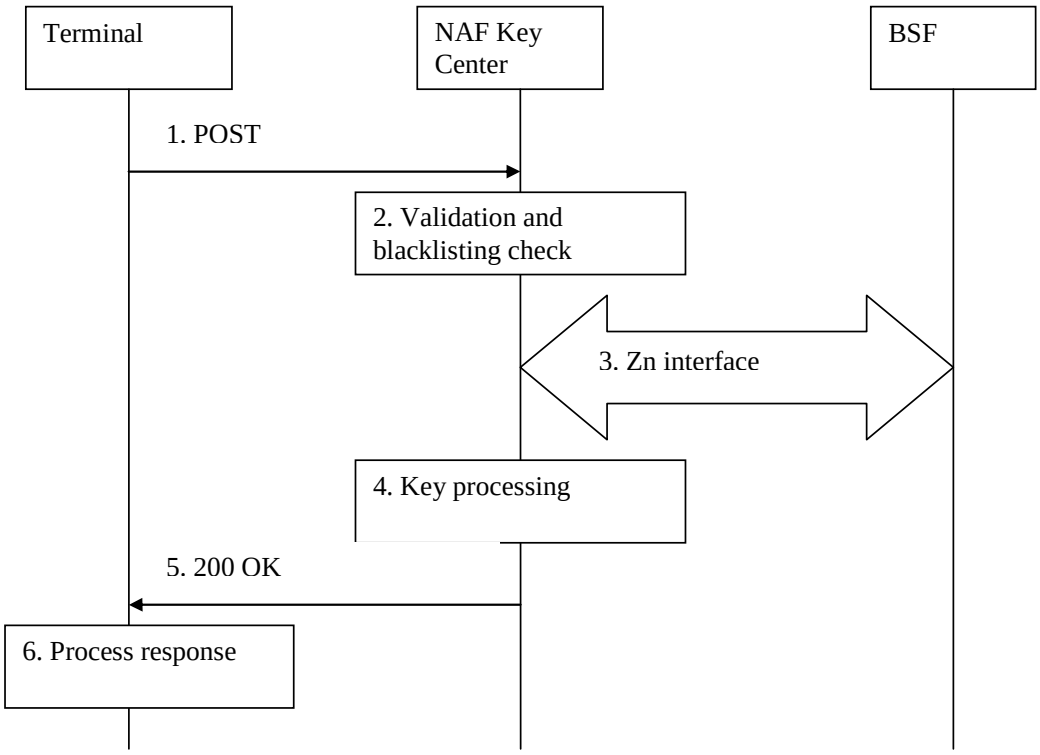

Figure D.2-1: Successful key request procedure between Terminal and NAF Key Center

1. Key request (Terminal to NAF Key Center) - see example in table D.2-1

The Terminal sends an HTTP request to the NAF Key Center containing a Key request.

Table D.2-1: Key request (Terminal to NAF Key Center)

```
POST /keyestablishment?requesttype=key-request-UICCkey  HTTP/1.1
Host: nafkeyCenter.home1.net:1234
Content-Type: application/keyest-UICCkeyrequest+xml
Content-Length: (...)
User-Agent: KeyestAgent; Release-7
Date: Thu, 27 June 2007 10:50:35 GMT
Accept: */*
Referrer: http://nafkeyCenter.home1.net:1234/service

<Key request BLOB>
```

Request-URI: The Request-URI (the URI that follows the method name, "POST", in the first line) indicates the resource of this POST request. The Request-URI contains the parameter "requesttype" which is set to "key-request-UICCkey" to indicate to the NAFKey Center the desired request type, i.e. Terminal requests for a Ks_local key.

Host: Specifies the Internet host and port number of the NAF Key Center, obtained from the original URI given by referring resource.

Content-Type: Contains the media type "application/keyest-UICCkeyrequest+xml", i.e.Key request.

Content-Length: Indicates the size of the entity-body, in decimal number of OCTETs, sent to the recipient.

User-Agent: Contains information about the user agent originating the request and the release of it.

Date: Represents the date and time at which the message was originated.

Accept: Media types which are acceptable for the response.

Referrer: Allows the user agent to specify the address (URI) of the resource from which the URI for the NAF Key Center was obtained.

2. **Validation and blacklisting check**

   The NAF Key Center will also verify that the DNS name in the realm attribute matches the NAF Key Center hostname. If the conversation is taking place inside a server-authenticated TLS tunnel, the NAF Key Center will also verify that this DNS name is the same as that of the TLS server.

   The NAF Key Center verifies that the Terminal is authorized to use this service as described in clause 4.5.2. If the authorization succeeds, the incoming client-payload request is taken in for further processing.

   If the NAF Key Center does not have the NAF specific key material (Ks_int_NAF), then the NAF Key Center retrieves that and one or more user security setting (USS) from the BSF. For detailed signalling flows see 3GPP TS 29.109 [16].

3. **Zn: NAF Key Center specific key procedure**

   NAF Key Center retrieves the NAF specific key material and IMPI of the user.

   For detailed signalling flows see TS 29.109 [16].

4. **Key processing in the NAF Key Center**

   NAF Key Center further derives the key establishment specific key material Ks_local as specified in Annex A.

5. **Response indicating success (NAF Key Center to Terminal)** - see example in table D.2-2

   The NAF Key Center sends 200 OK response to the Terminal to indicate the success of the Key request.

Table D.2-2: Key response (NAF Key Center to Terminal)

```
POST /keyestablishment?requesttype=key-response-UICCkey  HTTP/1.1
Host: nafkeyCenter.home1.net:1234
Content-Type: application/keyest-UICCkeyresponse+xml
Content-Length: (...)
Date: Thu, 27 June 2007 10:50:35 GMT
Accept: */*
Referrer: http://nafkeyCenter.home1.net:1234/service

<Key request BLOB>
```

Request-URI: The Request-URI (the URI that follows the method name, "POST", in the first line) indicates the resource of this POST request. The Request-URI contains the parameter "requesttype" which is set to "key-response-UICCkey" to indicate to the Terminal the desired response type, i.e. responses with a Ks_local key.

Host: Specifies the Internet host and port number of the NAF Key Center, obtained from the original URI given by referring resource.

Content-Type: Contains the media type "application/keyest-UICCkeyresponse+xml", i.e.Key response.

Content-Length: Indicates the size of the entity-body, in decimal number of OCTETs, sent to the recipient.

Date: Represents the date and time at which the message was originated.

Accept: Media types which are acceptable for the response.

Referrer: Allows the user agent to specify the address (URI) of the resource from which the URI for the NAF Key Center was obtained.

6. **Process response at Terminal**

The Terminal receives the response and accepts the server-payload for further processing. The Terminal stores the Ks_local key, the Key Lifetime and B-TID locally.

Annex E (normative): XML schema for Key Request and Key Response

E.1 Introduction

This annex contains the XML schema which defines a format for the key request sent from the Terminal to the NAF Key Center requesting a Ks_local key and for the key response sent from the NAF Key Center to the Terminal containing the Ks_local key, according to the procedures in section 4.5.2.

E.2 Key Request Format

E.2.1 Data Format

The below XML schema defines a format used to request a Ks_local key from the NAF Key Center.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns="urn:3GPP:metadata:2005:Keyest:UICCKeyRequest"
    targetNamespace="urn:3GPP:metadata:2005:Keyest:UICCKeyRequest"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

    <xs:element name="keyestUICCKeyRequest">
        <xs:annotation>
            <xs:documentation>
                Keyest Key Request as defined by 3GPP TS 33.110
            </xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="BTID" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="TERMINALID" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="TERMINALAPPLIID" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="UICCAPPLIID" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="RANDX" type="xs:string" minOccurs="1" maxOccurs="1"/>
            </xs:sequence>
            <xs:anyAttribute processContents="skip"/>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

E.2.2 Example

The below example is used to request a new Ks_local key derived from a GBA key with identity BTID "jhg876jhg", a TERMINAL ID "64783934857", TERMINALAPPLIID "7864934848", UICCAPPLIID "7864934849" and RANDX "12259673".

```
<?xml version="1.0" encoding="UTF-8"?>
<keyestUICCKeyRequest
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="urn:3GPP:metadata:2005:Keyest:UICCKeyRequest">
    <BTID>jhg876jhg</BTID>
    <TERMINALID>64783934857</TERMINALID>
    <TERMINALAPPLIID>7864934848</TERMINALAPPLIID>
    <UICCAPPLIID>7864934849</UICCAPPLIID>
    <RANDX>12259673</RANDX>
</keyestUICCKeyRequest>
```

E.3 Key Response Format

E.3.1 Data Format

The below XML schema defines a format used in the response to a Key request from the Terminal according to the procedure in section 4.5.2.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns="urn:3GPP:metadata:2005:Keyest:UICCKeyResponse"
    targetNamespace="urn:3GPP:metadata:2005:Keyest:UICCKeyResponse"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">

    <xs:element name="keyestUICCKeyResponse">
        <xs:annotation>
            <xs:documentation>
                Keyest Key Response as defined by 3GPP TS 33.110
            </xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:sequence>
                <xs:element name="BTID" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="KSLOCAL" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="KEYLIFETIME" type="xs:string" minOccurs="1" maxOccurs="1"/>
                <xs:element name="COUNTERLIMIT" type="xs:string" minOccurs="1" maxOccurs="1"/>
            </xs:sequence>
            <xs:anyAttribute processContents="skip"/>
        </xs:complexType>
    </xs:element>

</xs:schema>
```

E.3.2 Example

The below example is used in the key response from the NAF Key Center with identity BTID "jhg876jhg", KSLOCAL "64783934857", KEYLIFETIME "5675" and COUNTERLIMIT "3443".

```
<?xml version="1.0" encoding="UTF-8"?>
<keyestDeviceKeyResponse
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="urn:3GPP:metadata:2005:Keyest:DeviceKeyResponse">
    <BTID>jhg876jhg</BTID>
    <KSLOCAL>64783934857</KSLOCAL>
    <KEYLIFETIME>5675</KEYLIFETIME>
    <COUNTERLIMIT>3443</COUNTERLIMIT>

</keyestUICCKeyResponse>
```

Annex F (normative): TLS profiles

This annex provides the TLS profiles.

F.1 TLS profile for certificate based mutual authentication between Terminal and NAF Key Center

F.1.1 Introduction

The Terminal and the NAF Key Center shall support the TLS version as specified in RFC 2246 [5] or higher. Earlier versions are not allowed.

NOTE 1: The management of Root Certificates is out of scope of this Technical Specification.

The Terminal and the NAF Key Center shall support the server_name TLS extension. All other TLS extensions as specified in RFC 3546 [6] are optional for implementation.

F.1.2 Protection mechanisms

The Terminal shall support the CipherSuite TLS_RSA_ WITH_3DES_EDE_CBC_SHA and the CipherSuite TLS_RSA_WITH_AES_128_CBC_SHA. All other Cipher Suites as defined in RFC 2246 [5] and RFC 3268 [18] are optional for implementation for the Terminal.

The NAF Key Center shall support the CipherSuite TLS_RSA_ WITH_3DES_EDE_CBC_SHA the CipherSuite and the CipherSuite TLS_RSA_WITH_AES_128_CBC_SHA. All other Cipher Suites as defined in RFC 2246 [5] and RFC 3268 [18] are optional for implementation for the NAF Key Center.

Cipher Suites with NULL encryption are not allowed for use.

Cipher Suites with NULL integrity protection (or HASH) are not allowed for use.

F.2 TLS profile for Shared key-based mutual authentication between Terminal and NAF Key Center

F.2.1 Introduction

If the PSK TLS based authentication mechanism is supported, the HTTPS client in the Terminal or the NAF Key Center shall support the TLS version as specified in RFC 2246 [5], PSK TLS [19], or higher. Earlier versions are not allowed.

The HTTPS client in the Terminal and the NAF Key Center shall support the server_name TLS extension. All other TLS extensions as specified in RFC 3546 [6] are optional for implementation.

F.2.2 Protection mechanisms

The Terminal shall support the CipherSuite TLS_PSK_WITH_3DES_EDE_CBC_SHA and the CipherSuite TLS_PSK_WITH_AES_128_CBC_SHA. All other Cipher Suites as defined in PSK TLS [19] are optional for implementation for the Terminal.

The NAF Key Centre shall support the CipherSuite TLS_PSK_WITH_3DES_EDE_CBC_SHAand the CipherSuite TLS_PSK_WITH_AES_128_CBC_SHA. All other Cipher Suites as defined in PSK TLS [19] are optional for implementation for the NAF Key Center.

Cipher Suites with NULL encryption are not allowed for use.

Cipher Suites with NULL integrity protection (or HASH) are not allowed for use.

Annex G (informative): Change history

Change history									
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Cat**	**Subject/Comment**	**Old**	**New**	**WI**
2006-01						Creation of document		0.0.0	
2006-05						Integration of pseudo-CRs S3-060265, S3-060280, S3-060282, and creation of annex based on contributions S3-060258 and S3-060309.	0.0.0	0.1.0	KeyEstUTerm
2006-07	SP-33					Integration of pseudo-CRs S3-060432, S3-060468, S3-060469 and S3-060569	0.1.0	1.0.0	KeyEstUTerm
2006-11	SP-34					Integration of pseudo-CRs S3-060669, S3-060672, S3-060673, S3-060674, S3-060754.	1.0.0	2.0.0	KeyEstUTerm
2006-12	SP-34	SP-060807	-	-	-	Approved at SA #34	2.0.0	7.0.0	KeyEstUTerm
2007-03	SP-35	SP-070155	0001	-	C	NAF Key Center shall authorize/administrate Terminal_appl_ID and UICC_appl_ID	7.0.0	7.1.0	KeyEstUTerm
2007-03	SP-35	SP-070155	0003	-	F	Figure 4-3 misleadingly lists Ks_NAF in message 9	7.0.0	7.1.0	KeyEstUTerm
2007-03	SP-35	SP-070155	0004	1	F	Keep annex alignment with the specification text	7.0.0	7.1.0	KeyEstUTerm
2007-06	SP-36	SP-070330	0006	-	F	Addition of reference to GAA Service Type Code	7.1.0	7.2.0	KeyEstUTerm
2007-06	SP-36	SP-070330	0007	1	F	Addition of annex on key establishment UICC-Terminal interface	7.1.0	7.2.0	KeyEstUTerm
2007-06	SP-36	SP-070330	0009	1	C	Addition of key confirmation and various other changes	7.1.0	7.2.0	KeyEstUTerm
2007-09	SP-37	SP-070600	0010	1	F	Ks_local keys storage policy in the UICC	7.2.0	7.3.0	KeyEstUTerm
2007-09	SP-37	SP-070600	0012	1	F	Complete NAF Key Center procedure to check Terminal_ID.	7.2.0	7.3.0	KeyEstUTerm
2007-09	SP-37	SP-070600	0011	1	F	Stage 3 details : Ua interface and adding TLS Profiling	7.2.0	7.3.0	KeyEstUTerm
2007-09	SP-37	SP-070600	0013	1	F	Addition of PSK-TLS to secure communication between the Terminal and the NAF Key Center.	7.2.0	7.3.0	KeyEstUTerm
2007-12	SP-38	SP-070889	0014	3	F	Clarification of payload in 'service request' message	7.3.0	7.4.0	KeyEstUTerm
2007-12	SP-38	SP-070889	0015	3	F	Correction of XML schema	7.3.0	7.4.0	KeyEstUTerm
2008-03	SP-39	SP-080148	0016			Removal of editor"s notes	7.4.0	7.5.0	KeyEstUTerm
2008-12	SP-42	SP-080746	0017	1	F	Removing editor's note on IANA registration	7.5.0	8.0.0	TEI8

History

Document history		
V8.0.0	January 2009	Publication