CR-Form-v7.1

CHANGE REQUEST

| ⌘ | **33.220** CR **041** | ⌘ **rev** | **1** | ⌘ | Current version: | **6.2.0** | ⌘ |

*For **HELP** on using this form, see bottom of this page or look at the pop-up text over the ⌘ symbols.*

Proposed change affects: | UICC apps⌘ **X**   ME **X** Radio Access Network ☐ Core Network **X**

Title:	⌘	Key derivation function

Source:	⌘	SA WG3

Work item code:	⌘	SEC1-SC		**Date:** ⌘	16/11/2004

Category:	⌘	**B**			**Release:** ⌘	Rel-6

Use one of the following categories:
    F (correction)
    A (corresponds to a correction in an earlier release)
    B (addition of feature),
    C (functional modification of feature)
    D (editorial modification)
Detailed explanations of the above categories can be found in 3GPP TR 21.900.

Use one of the following releases:
    Ph2   (GSM Phase 2)
    R96   (Release 1996)
    R97   (Release 1997)
    R98   (Release 1998)
    R99   (Release 1999)
    Rel-4 (Release 4)
    Rel-5 (Release 5)
    Rel-6 (Release 6)
    Rel-7 (Release 7)

Reason for change:	⌘	SAGE has defined a key derivation function (KDF) in their LS to SA3 (S3-040914 updated in S3-040937). This CR implements the KDF to TS 33.220.

Summary of change:	⌘	Annex B contains the KDF as specified by SAGE. Subclauses 4.2.1 and 4.5.2 refer to Annex B.

Consequences if not approved:	⌘	The KDF is not specified.

Clauses affected:	⌘	2, 4.2.1, 4.5.2, Annex B

		Y	**N**		
Other specs affected:	⌘		X	Other core specifications	⌘
			X	Test specifications	
			X	O&M Specifications	

Other comments:	⌘	

===== **BEGIN CHANGE** =====

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 TS 31.102: "3rd Generation Partnership Project; Technical Specification Group Terminals; Characteristics of the USIM application".

[2]   3GPP TS 33.102: "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Security architecture".

[3]   Franks J., et al,: "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.

[4]   A. Niemi, et al,: "Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA)", RFC 3310, September 2002.

[5]   3GPP TS 33.221: "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Generic Authentication Architecture (GAA); Support for Subscriber Certificates".

[6]   T. Dierks, et al,: "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[7]   OMA: "Provisioning Content Version 1.1", Version 13-Aug-2003. Open Mobile Alliance.

[8]   3GPP TS 23.228: "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; IP Multimedia Subsystem (IMS); Stage 2 (Release 6)".

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

[10]   3GPP TS 31.103: "3rd Generation Partnership Project; Technical Specification Group Terminals; Characteristics of the IP Multimedia Services Identity Module (ISIM) application".

[11]   3GPP TS 23.003: "3rd Generation Partnership Project; Technical Specification Group Core Network; Numbering, addressing and identification".

[12]   IETF RFC 3548 (2003): "The Base16, Base32, and Base64 Data Encodings".

[13]   3GPP TS 33.210: "3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Network domain security; IP network layer security".

[14]   IETF RFC 3588 (2003): "Diameter Base Protocol".

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

[16]   ISO/IEC 10118-3:2004 Information Technology – Security techniques – Hash-functions – Part 3: Dedicated hash-functions

===== **BEGIN NEXT CHANGE** =====

4.2.1 Bootstrapping server function (BSF)

A generic Bootstrapping Server Function (BSF) and the UE shall mutually authenticate using the AKA protocol, and agree on session keys that are afterwards applied between UE and an operator-controlled Network Application Function (NAF). The BSF shall restrict the applicability of the key material to a specific NAF by using a suitablethe key derivation procedure as specified in Annex B. The key derivation procedure may be used with multiple NAFs during the lifetime of the key material. The lifetime of the key material is set according to the local policy of the BSF. The generation of key material is specified in clause 4.5.2.

The BSF shall be able to acquire the GBA user security settings from the HSS.

===== **BEGIN NEXT CHANGE** =====

4.5.2 Bootstrapping procedures

When a UE wants to interact with a NAF, and it knows that the bootstrapping procedure is needed, it shall first perform a bootstrapping authentication (see figure 4.3). Otherwise, the UE shall perform a bootstrapping authentication only when it has received bootstrapping initiation required message or a bootstrapping negotiation indication from the NAF, or when the lifetime of the key in UE has expired (cf. subclause 4.5.3).

NOTE 1: The main steps from the specifications of the AKA protocol in TS 33.102 [2] and the HTTP digest AKA protocol in RFC 3310 [4] are repeated in figure 3 for the convenience of the reader. In case of any potential conflict, the specifications in TS 33.102 [2] and RFC 3310 [4] take precedence.
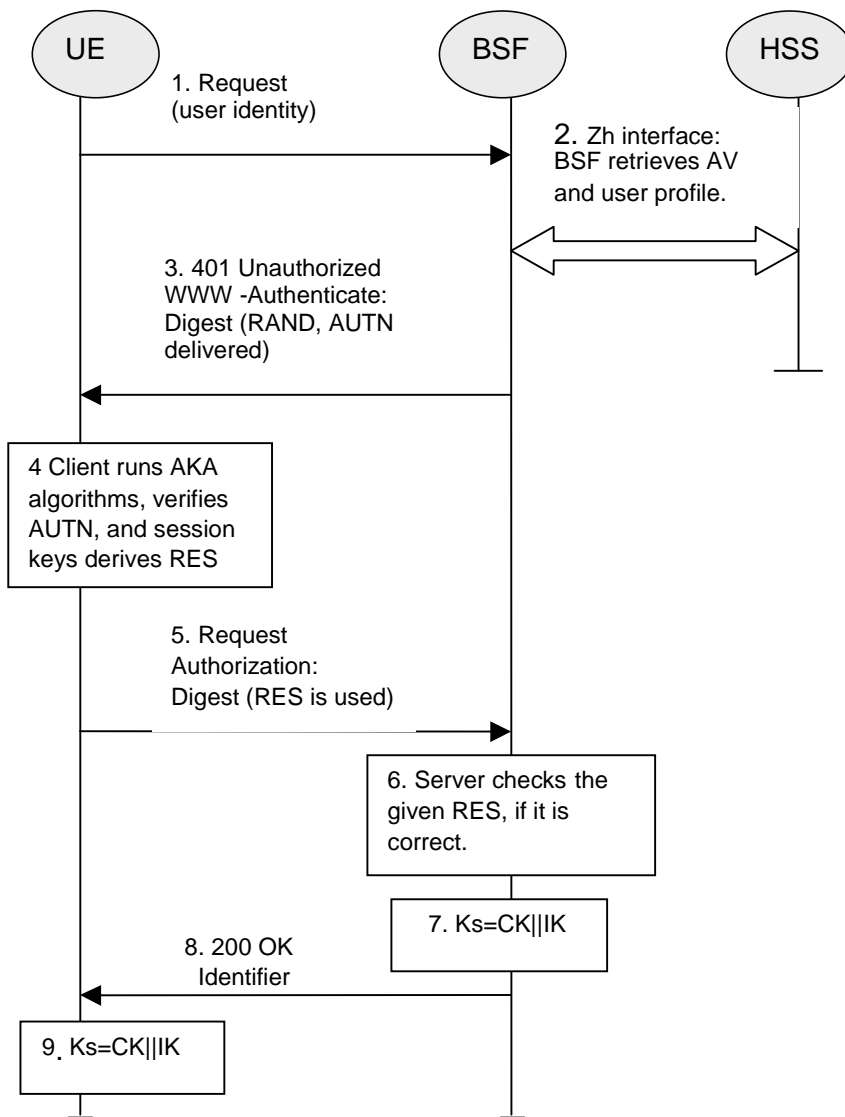
Figure 4.3: The bootstrapping procedure

1. The UE sends an HTTP request towards the BSF.

2. BSF retrieves the complete set of GBA user security settings and one or a whole batch of Authentication Vectors (AV, AV = RAND‖AUTN‖XRES‖CK‖IK) over the reference point Zh from the HSS.

3. Then BSF forwards the RAND and AUTN to the UE in the 401 message (without the CK, IK and XRES). This is to demand the UE to authenticate itself.

4. The UE checks AUTN to verify that the challenge is from an authorised network; the UE also calculates CK, IK and RES. This will result in session keys IK and CK in both BSF and UE.

5. The UE sends another HTTP request, containing the Digest AKA response (calculated using RES), to the BSF.

6. The BSF authenticates the UE by verifying the Digest AKA response.

7. The BSF generates key material Ks by concatenating CK and IK. The B-TID value shall be also generated in format of NAI by taking the base64 encoded [12] RAND value from step 3, and the BSF server name, i.e. base64encode(RAND)@BSF_servers_domain_name.

8. The BSF shall send a 200 OK message, including a B-TID, to the UE to indicate the success of the authentication. In addition, in the 200 OK message, the BSF shall supply the lifetime of the key Ks. The key material Ks is generated in UE by concatenating CK and IK.

9. Both the UE and the BSF shall use the Ks to derive the key material Ks_NAF during the procedures as specified in clause 4.5.3. Ks_NAF shall be used for securing the reference point Ua.

Ks_NAF is computed as Ks_NAF = KDF (Ks, "gba-me" || RAND || IMPI || NAF_Idkey derivation parameters), where KDF is a suitablethe key derivation function as specified in Annex B, and the key derivation parameters consist of the user's IMPI, the NAF_Id and RAND. The NAF_Id consists of the full DNS name of the NAF. KDF shall be implemented in the ME.

NOTE 2: To allow consistent key derivation based on NAF name in UE and BSF, at least one of the three following prerequisites shall be fulfilled:

(1) The NAF is known in DNS under one domain name (FQDN) only, i.e. no two different domain names point to the IP address of the NAF. This has to be achieved by administrative means.
This prerequisite is not specific to 3GPP, as it is necessary also under other circumstances, e.g. for TLS V1.0 without use of wildcard or multiple-name certificates.

(2) Each DNS entry of the NAF points to a different IP address. The NAF responds to all these IP addresses. Each IP address is tied to the corresponding FQDN by NAF configuration. The NAF can see from the IP address, which FQDN to use for key derivation.

(3) Ua uses a protocol which transfers the host name (FQDN of NAF as used by UE) to NAF (e.g. HTTP/1.1 with mandatory Host request header field). This requires the NAF to check the validity of the host name, to use this name in all communication with UE where appropriate, and to transfer this name to BSF to allow for correct derivation of Ks_NAF.
In case of a TLS tunnel this requires either multiple-identities certificates or the deployment of RFC 3546 [9] or other protocol means with similar purpose.

Editor's note: The definition of the KDF is left to ETSI SAGE and is to be included in the Annex B of the present specification.

The UE and the BSF shall store the key Ks with the associated B-TID for further use, until the lifetime of Ks has expired, or until the key Ks is updated.

===== **BEGIN NEXT CHANGE** =====

Annex B (normative):
Specification of the key derivation function KDF

Editor's note: The definition of the KDF and the possible inclusion of further key derivation parameters is left to ETSI SAGE.

B.1   Introduction

This annex specifies the key derivation function (KDF) that is used in the NAF specific key derivation in both GBA (i.e., GBA_ME) and GBA_U. The key derivation function defined in the annex takes the following assumptions:

1. the input parameters to the key derivation functions are octet strings - not bit strings of arbitrary length:

2. a single input parameter will have lengths no greater than 65535 octets.

B.2   Generic key derivation function

The input parameters and their lengths shall be concatenated into a string S as follows:

1. The length of each input parameter in octets shall be encoded into two-octet string:

    a) express the number of octets in input parameter Pi as a number l in the range $0 \leq l \leq 65535$.

    b) Li is then a two-octet representation of the number l, with the most significant bit of the first octet of Li equal to the most significant bit of l, and the least significant bit of the second octet of Li equal to the least significant bit of l,

    Example: If Pi contains 258 octets then Li will be the two-octet string 0x01 0x02.

2. String S shall be constructed from n input parameters as follows:

    S = FC || P0 || L0 || P1 || L1 || P2 || L2 || P3 || L3 ||... || Pn || Ln

    where

       FC is single octet used to distinguish between different instances of the algorithm,

       P0 is a static ASCII-encoded string,

       L0 is the two octet representation of the length of the P0,

       P1 ... Pn are the n input parameters, and

       L1 ... Ln are the two-octet representations of the corresponding input parameters.

3. The final output, i.e., the derived key is equal to HMAC-SHA-256 (as specified in [15] and [16]) computed on the string S using the key Key:

    derived key = HMAC-SHA-256 (Key , S)

B.3   NAF specific key derivation in GBA and GBA_U

In GBA and GBA_U, the input parameters for the key derivation function shall be the following:

-    FC = 0x01,

-    P1 = RAND,

- L1 = length of RAND is 16 octets (i.e., 0x00 0x10),

- P2 = IMPI,

- L2 = length of IMPI is variable (not greater that 65535),

- P3 = NAF_ID, and

- L3 = length of NAF_ID is variable (not greater that 65535).

In the key derivation of Ks_NAF as specified in clause 4 and Ks_ext_NAF as specified in clause 5,

- P0 = "gba-me" (i.e., 0x67 0x62 0x61 0x2d 0x6d 0x65), and

- L0 = length of P0 is 6 octets (i.e., 0x00 0x06).

In the key derivation of Ks_int_NAF as specified in clause 5,

- P0 = "gba-u" (i.e., 0x67 0x62 0x61 0x2d 0x75), and

- L0 = length of P0 is 5 octets (i.e., 0x00 0x05).

The Key to be used in key derivation shall be:

- Ks (i.e., CK || IK concatenated) as specified in clauses 4 and 5,

NOTE: In the specification this function is denoted as:
Ks_NAF = KDF (Ks, "gba-me" || RAND || IMPI || NAF_Id),
Ks_ext_NAF = KDF (Ks, "gba-me" || RAND || IMPI || NAF_Id), and
Ks_int_NAF = KDF (Ks, "gba-u" || RAND || IMPI || NAF_Id).

===== END CHANGE =====