**3GPP TSG-SA WG3 Meeting S3#35**
**St Paul's Bay, Malta, October 5-8, 2004**

***Tdoc ⌘S3-040783***

*CR-Form-v7.1*

# CHANGE REQUEST

⌘ **33.220** **CR** **028** ⌘ **rev** **-** ⌘ 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:*** | ⌘ | Enabling optional GBA_U support for ME |
| ***Source:*** | ⌘ | Nokia, Siemens, Ericsson, Samsung Electronics |
| ***Work item code:*** | ⌘ | SEC1-SC |
| ***Date:*** | ⌘ | 27/09/2004 |
| ***Category:*** | ⌘ | **C** |
| ***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:*** | ⌘ | The details of GBA_U and GBA_ME based key derivations are missing. |
| ***Summary of change:*** | ⌘ | 1. GBA_U support for GBA enabled Rel6 MEs is optional.<br>2. GBA_U and GBA_ME key derivation details are defined.<br>3. Ks_ext_NAF and Ks_int_NAF are 256 bits in length. |
| ***Consequences if not approved:*** | ⌘ | The details of GBA_U and GBA_ME based key derivations are missing. |

| | | |
|---|---|---|
| ***Clauses affected:*** | ⌘ | 5.2.1, 5.3.2 |

| ***Other specs affected:*** | ⌘ | Y | N | | ⌘ | |
|---|---|---|---|---|---|---|
| | | X | | Other core specifications | ⌘ | TS 31.102, TS 33.103 |
| | | | X | Test specifications | | |
| | | | X | O&M Specifications | | |

| | | |
|---|---|---|
| ***Other comments:*** | ⌘ | |

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

## 5.2.1 Requirements on UE

The 3G AKA keys CK and IK resulting from a run of the protocol over the Ub reference point shall not leave the UICC.

The UICC shall be able to distinguish between authentication requests for GBA_U, and authentication requests for other 3G authentication domains.

Upon an authentication request (i.e., AUTHENTICATE command) from the ME, which the UICC recognises as related to GBA_U, the UICC shall derive ~~two~~four keys from CK and IK called CK', IK', CK" and IK". The length of CK', IK', CK" and IK" shall be 128 bits. All 3G MEs are capable of such a request.

Editor's note: The definition of exact derivation of CK', IK', CK" and IK" is left to ETSI SAGE.

The UICC shall deliver CK' and IK' to the ME in response to an AUTHENTICATE command as defined in TS 31.102 [1]. The ME shall derive the Ks_ext by concatenating CK' and IK' (i.e., Ks_ext = CK' || IK').

NOTE: As the number and the meaning of the parameters to the AUTHENTICATE command are the same in this GBA_U security context as in 3G Release 5 specifications, the GBA_U unaware ME may transparently use the GBA_U aware UICC as it would use a GBA_U unaware UICC.

The UICC shall derive the Ks_int by concatenating CK" and IK" (i.e., Ks_int = CK" || IK"), and store it to the UICC.

Upon request from the ME, the UICC shall be able to derive further NAF-specific keys (Ks_int_NAF) from the derived key stored on the UICC (Ks_int). Only GBA_U-aware 3G MEs are capable of such a request.

~~Editors' Note: The location (whether in the UICC or in the ME) of the storage of Ks_ext is ffs.~~

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

## 5.3.2 Bootstrapping procedure

The procedure specified in this clause differs from the procedure specified clause 4.5.2 in the local handling of keys and Authentication Vectors in the UE and the BSF. The messages exchanged over the Ub reference point are identical for both 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 5.1). Otherwise, the UE shall perform a bootstrapping authentication only when it has received bootstrapping initiation required message or a bootstrapping renegotiation indication from the NAF, or when the lifetime of the key in UE has expired (see clause 5.3.3).

NOTE: 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 5.1 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.

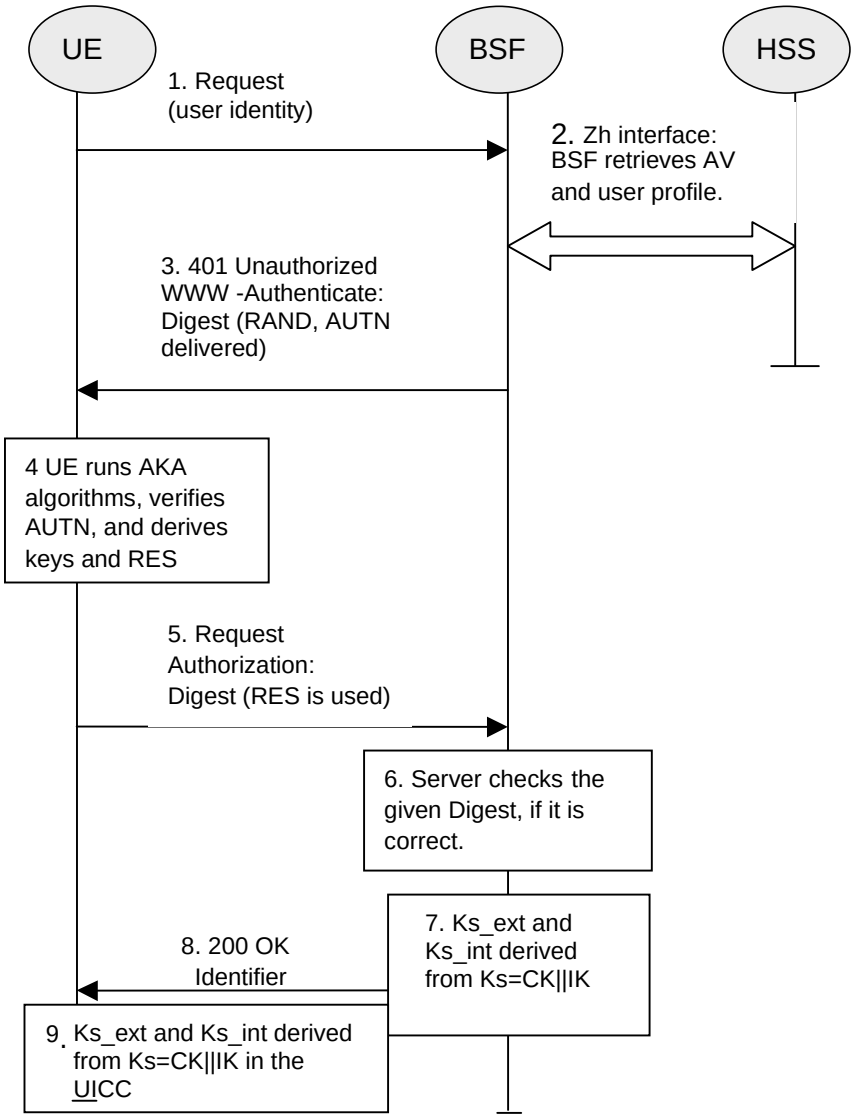


**Figure 5.1: The bootstrapping procedure with UICC-based enhancements**

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

2. The 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 Zh reference point from the HSS. The BSF can then decide to perform GBA_U, based on the user security settings (USSs). In this case, the BSF proceeds in the following way:

- BSF computes MAC* = MAC SHA-1(IK1) (where IK= IK1|| IK2 and * is a exclusive or as described in TS 33.102 [2])

Editor's note: The exact format of the MAC modification function is to be reviewed. The output of SHA-1 needs to be truncated to exact amount of bits needed (64 bits).

The BSF stores the XRES after flipping the least significant bit.

3. Then BSF forwards the RAND and AUTN* (where AUTN* = SQN ⊕ AK || AMF || MAC*) to the UE in the 401 message (without the CK, IK and XRES). This is to demand the UE to authenticate itself.

4. The ME sends RAND and AUTN* to the UICC. The UICCcalculates IK and MAC (by performing MAC= MAC* ⊕ SHA-1(IK1 )). Then the UICC checks AUTN(i.e. SQN ⊕ AK || AMF || MAC) to verify that the

challenge is from an authorised network; the UICC also calculates CK and RES. This will result in session keys CK and IK in both BSF and UICC.

5. The UICC then ~~applies a suitable key derivation function h1 to Ks, which is the concatenation of CK and IK, and possibly further h1-key derivation parameters to obtain two keys, Ks_ext and Ks_int, each of length 128 bit, i.e. h1(Ks, h1-key derivation parameters) = Ks_ext || Ks_int (see also figure 5.2)~~ derives four keys CK', IK', CK", and IK" by applying a suitable key derivation function h1 to CK, IK, and other possible key derivation parameters. The UICC generates the Ks_int by concatenating CK" and IK". The UICC then transfers RES (after flipping the least significant bit) ~~and Ks_ext~~, IK', and CK' to the ME and stores Ks_int ~~/ks_ext~~ on the UICC. The ME generates Ks_ext by concatenating CK' and IK'. See also figure 5.2.

Editors' Note: The definition of the h1 is left to ETSI SAGE and is to be included in the Annex B of the present specification.

~~Editors' Note: The location (whether in the UICC or in the ME) of the storage of Ks_ext is ffs.~~

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

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

8. The BSF generates the key Ks by concatenating CK and IK. Then the BSF applies the key derivation function h1 to ~~Ks and possibly further h1-key derivation parameters to obtain two keys,~~ CK and IK as the UICC did in step 5 to derive CK', IK', CK", and IK". Ks_ext and Ks_int~~,~~ are generated in the same way as the UICC did in step 5. 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.

9. The BSF shall send a 200 OK message, including the 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 keys Ks_ext and Ks_int, The lifetimes of the keys Ks_ext and Ks_int shall be the same.

10. The BSF shall use the keys Ks_ext and Ks_int to derive the NAF-specific keys Ks_ext_NAF and Ks_int_NAF, each of length 256 bits, if requested by a NAF over the Zn reference point. Ks_ext_NAF and Ks_int_NAF are used for securing the Ua reference point. The UE shall use the key Ks_ext to derive the NAF-specific key Ks_ext_NAF, if applicable. The UICC shall use the key Ks_int to derive the NAF-specific key Ks_int_NAF, if applicable.

    Ks_ext_NAF is computed as Ks_ext_NAF = h2 (Ks_ext, h2-key derivation parameters), and Ks_int_NAF is computed in the UICC as Ks_int_NAF = h2 (Ks_int, h2-key derivation parameters), where h2 is a suitable key derivation function, and the h2-key derivation parameters include the user's IMPI, the NAF_Id and RAND. The NAF_Id consists of the full DNS name of the NAF.

Editors' Note: The definition of the h2 is left to ETSI SAGE and is to be included in the Annex B of the present specification.

NOTE: The NOTE 2 of clause 4.5.2 also applies here.

The ME, the UICC and the BSF store the keys Ks_ext and Ks_int together with the associated B-TID for further use, until the lifetime of Ks_ext and Ks_int has expired, or until the keys Ks_ext and Ks_int are updated.

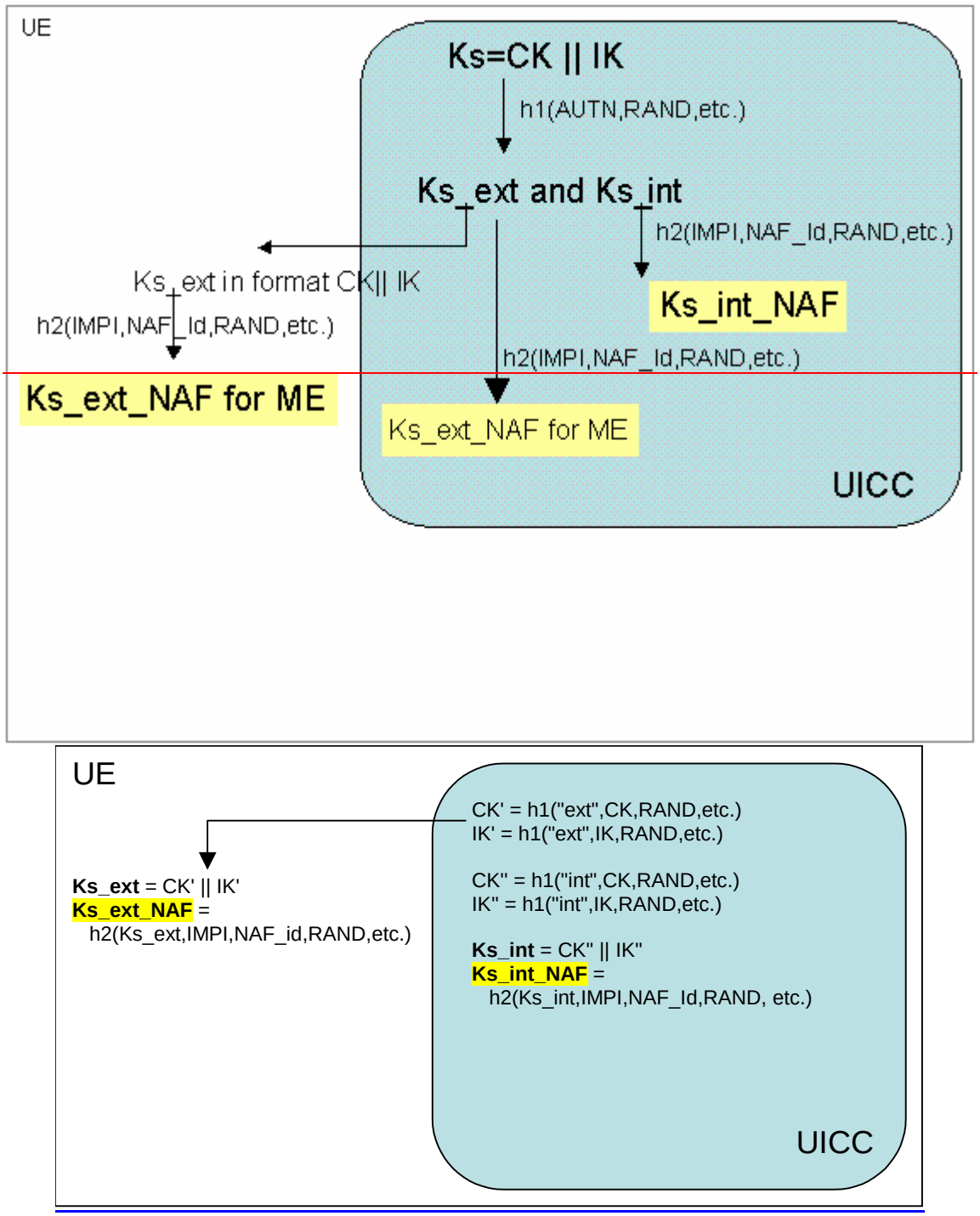


**Figure 5.2: Key derivation for GBA-aware UICC when GBA-run was triggered**

Editor's note: Figure 5.2 needs to be update after ETSI SAGE has defined the key derivation functions for GBA_U.

**===== END CHANGE =====**