3GPP TSG SA 3 Meeting #10
Antwerp, Belgium, 19-21 January 2000

***Document* S3-000078**
e.g. for 3GPP use the format TP-99xxx
or for SMG, use the format P-99-xxx

DRAFT CHANGE REQUEST

Please see embedded help file at the bottom of this page for instructions on how to fill in this form correctly.

33.105	CR	006	Current Version:	3.2.0

GSM (AA.BB) or 3G (AA.BBB) specification number ↑ *↑ CR number as allocated by MCC support team*

For submission to:	TSG SA #7	for approval	X	strategic		(for SMG use only)
list expected approval meeting # here ↑		for information		non-strategic		

Form: CR cover sheet, version 2 for 3GPP and SMG *The latest version of this form is available from: ftp://ftp.3gpp.org/Information/CR-Form-v2.doc*

Proposed change affects: *(at least one should be marked with an X)*	(U)SIM	X	ME		UTRAN / Radio		Core Network	X

Source: Siemens Atea **Date:** 21-01-00

Subject: Authentication and key agreement

Work item: Security

Category: *(only one category shall be marked with an X)*

F	Correction	X	**Release:**	Phase 2	
A	Corresponds to a correction in an earlier release			Release 96	
B	Addition of feature			Release 97	
C	Functional modification of feature			Release 98	
D	Editorial modification			Release 99	X
				Release 00	

Reason for change: Bring TS 33.105 in line with the decisions taken in TSG SA and the CRs to TS 33.102 that have been approved. This includes the replacement of MODE by AMF, the change as regards input parameters to the computation of the re-synchronisation token AUTS, the deletion of the alternative mechanism (annex B) and the deletion of annex C on unspecified values, as all these value have in the mean time been specified.

Clauses affected: 5.1, annex B, annex C

Other specs affected:

Other 3G core specifications		→ List of CRs:
Other GSM core specifications		→ List of CRs:
MS test specifications		→ List of CRs:
BSS test specifications		→ List of CRs:
O&M specifications		→ List of CRs:

Other comments:

help.doc

<--------- double-click here for help and instructions on how to create a CR.

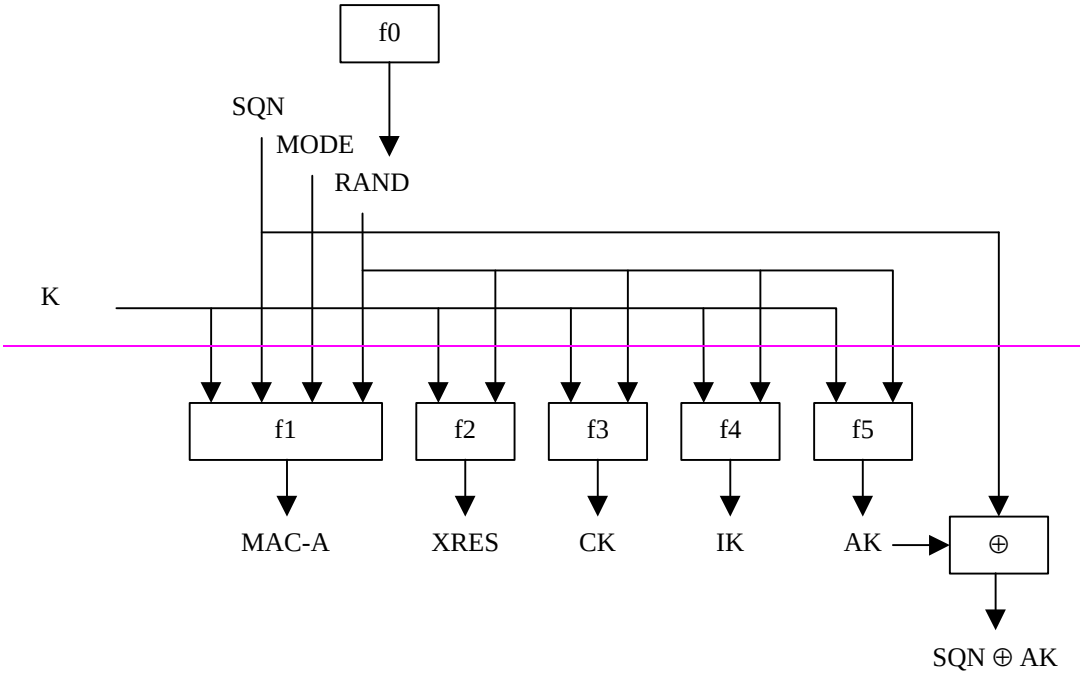
5.1 Authentication and key agreement

5.1.1 Overview

The mechanism for authentication and key agreement described in clause 6.3 of [1] requires the following cryptographic functions:

f0	the random challenge generating function;
f1	the network authentication function;
f1*	the re-synchronisation message authentication function;
f2	the user authentication function;
f3	the cipher key derivation function;
f4	the integrity key derivation function;
f5	the anonymity key derivation function.

~~Figure 1 illustrates the use of the cryptographic functions f0 – f5 at the AuC.~~



~~Figure 1: Functions for authentication and key agreement in the AuC~~

~~The input parameters to the algorithms are the long-term secret Key (K), the Sequence Number (SQN), a random challenge (RAND) generated by f0 and a mobility management mode indicator (MODE). Based on K and RAND f2 – f5 compute an expected RESponse (XRES), a Cipher Key (CK), an Integrity Key (IK) and an Anonymity Key (AK) which is bitwise XOR-ed with SQN. Based on K, RAND, SQN and MODE, f1 derives a Message Authentication Code for network to user Authentication (MAC-A).~~

~~Figure 2 illustrates the use of function f1* in the AuC .~~

~~The input parameters to the algorithm f1* are the long-term secret Key (K), the Sequence Number (SQN), a random challenge (RAND) received by the AuC from the SN and a mobility management mode indicator (MODE). Based on these input parameters, an expected message authentication code XMAC-S on the synchronisation failure data is computed.~~

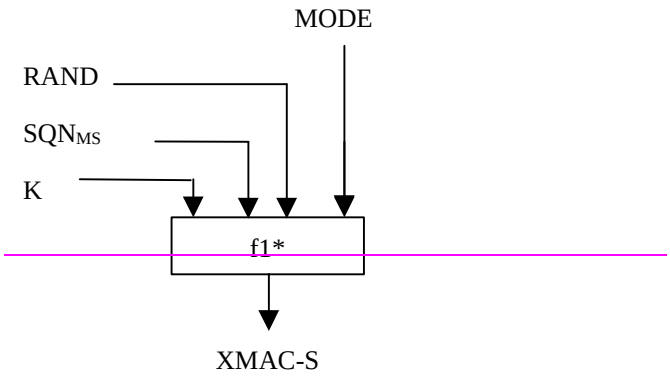


Figure 2: The re-synchronisation message authentication function in the AuC

Figure 3 illustrates the use of the cryptographic functions f1 – f5 in the USIM.

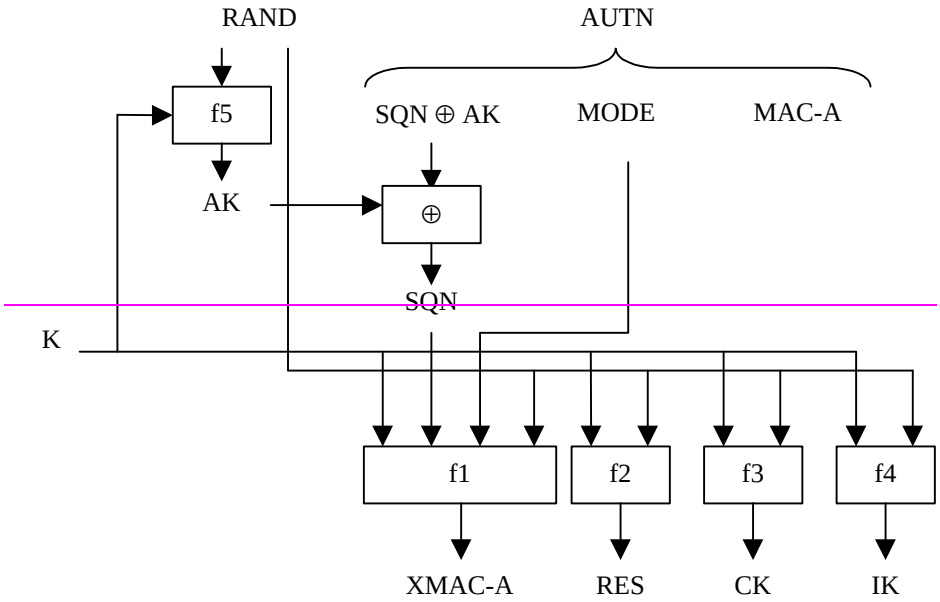


Figure 3: Functions for authentication and key agreement in the USIM

This time the Anonymity Key (AK) has to be computed in advance to retrieve SQN from SQN ⊕ AK. From there on, f1 is used to derive XMAC-A from K, RAND, SQN and MODE and f2 – f4 are used to derive RES, CK and IK in the same way as the AuC derived MAC-A, XRES, CK and IK.

The use of the function in f1* in the USIM is strictly analogous to that in the AuC. A figure is therefore omitted. Based on the input parameters long-term secret Key (K), Sequence Number (SQN), random challenge (RAND) received by the USIM from the SN and mobility management mode indicator (MODE), a message authentication code MAC-S on synchronisation failure data is computed.

NOTE: An alternative mechanism for authentication and key agreement that requires a different set of cryptographic functions is included in Annex D of [1]. The requirements on the cryptographic functions required for that mechanism are described in Annex A of this specification.

5.1.1.1 Generation of quintets in the AuC

To generate a quintet the HLR/AuC

– computes a message authentication code for authentication MAC-A = $f1_K$(SQN || RAND || AMF), an expected response XRES = $f2_K$ (RAND), a cipher key CK = $f3_K$ (RAND) and an integrity key IK = $f4_K$ (RAND) where f4 is a key generating function.

– If SQN is to be concealed, in addition the HLR/AuC computes an anonymity key AK = $f5_K$ (RAND) and computes the concealed sequence number SQN ⊕ AK = SQN xor AK. Concealment of the sequence number is optional.

– Finally, the HLR/AuC assembles the authentication token AUTN = SQN [⊕ AK] || AMF || MAC-A and the quintet Q = (RAND, XRES, CK, IK, AUTN).

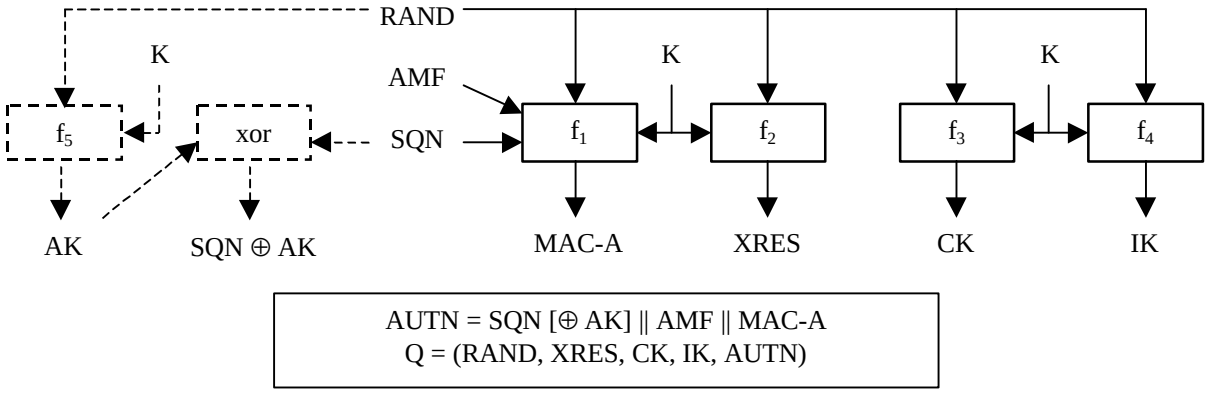


Figure 1: Generation of quintets in the AuC

5.1.1.2 Authentication and key derivation in the USIM

Upon receipt of a (RAND, AUTN) pair the USIM acts as follows:

– If the sequence number is concealed, the USIM computes the anonymity key AK = $f5_K$(RAND) and retrieves the unconcealed sequence number SQN = (SQN ⊕ AK) xor AK.

The USIM computes XMAC-A = $f1_K$ (SQN || RAND || AMF), the response RES = $f2_K$(RAND), the cipher key CK = $f3_K$ (RAND) and the integrity key IK = $f4_K$ (RAND).

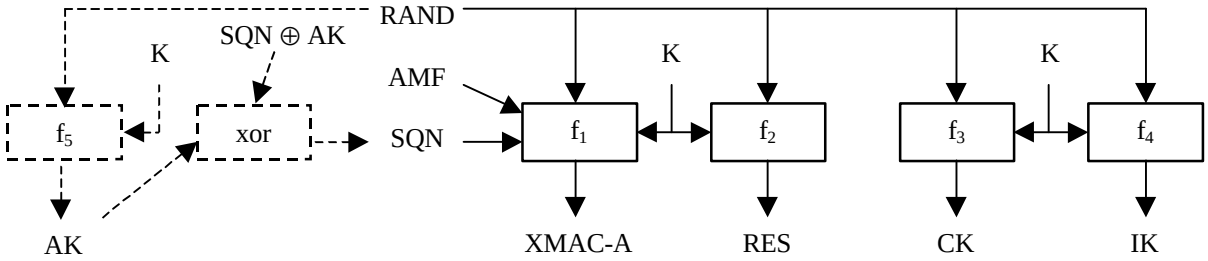


Figure 2: Authentication and key derivation in the USIM

5.1.1.3 Generation of re-synchronisation token in the USIM

Upon the assertion of a synchronisation failure, the USIM generates a re-synchronisation token as follows:

a) The USIM computes MAC-S = $f1^*_K$(SQN_{MS} || RAND || AMF*), whereby AMF* is a default value for AMF used in re-synchronisation.

b) If SQN_{MS} is to be concealed with an anonymity key AK, the USIM computes AK = $f5_K$(MAC-S || 0…0), whereby MAC-S forms the 12 most significant octets and 32 zeros form the 4 least significant octets of the required 16 octet input parameter, and the concealed counter value is then computed as SQN_{MS} ⊕ AK.

c) The re-synchronisation token is constructed as AUTS = SQN_{MS} [⊕ AK] || MAC-S.

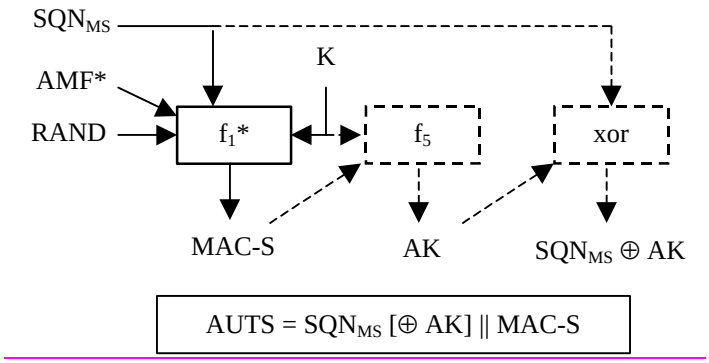


Figure 3: Generation of re-synchronisation token in the USIM

5.1.1.4 Re-synchronisation in the HLR/AuC

Upon receipt of an indication of synchronisation failure and a (AUTS, RAND) pair, the HLR/AuC may perform the following cryptographic functions:

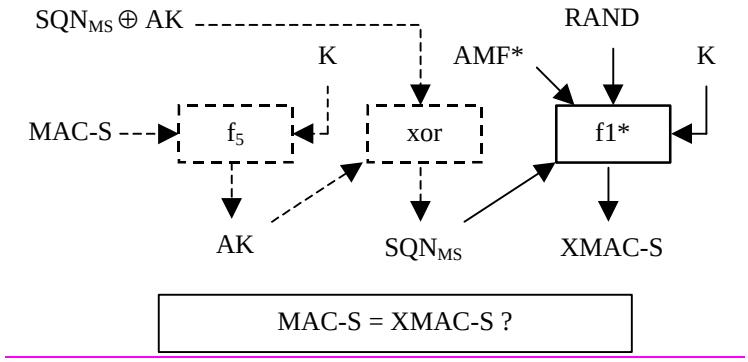


Figure 4: Re-synchronisation in the HLR/AuC

a) If SQN_{MS} is concealed with an anonymity key AK, the HLR/AuC computes $AK = f5_K(MAC\text{-}S \,||\, 0\ldots0)$, whereby MAC-S forms the 12 most significant octets and 32 zeros form the 4 least significant octets of the required 16 octet input parameter and retrieves the unconcealed counter value as $SQN_{MS} = (SQN_{MS} \oplus AK)$ xor AK.

b) If SQN generated from SQN_{HE} would not be acceptable, then the HLR/AuC computes $XMAC\text{-}S = f1^*_K(SQN_{MS} \,||\, RAND \,||\, AMF^*)$, whereby AMF* is a default value for AMF used in re-synchronisation.

5.1.2 Use

The functions f0—f5 shall only be used to provide mutual entity authentication between USIM and AuC, derive keys to protect user and signalling data transmitted over the radio access link and conceal the sequence number to protect user identity confidentiality. The function f1* shall only be used to provide data origin authentication for the synchronisation failure information sent by the USIM to the AuC.

5.1.3 Allocation

The functions f1—f5 and f1*are allocated to the Authentication Centre (AuC) and the USIM. The function f0 is allocated to the AuC.

5.1.4 Extent of standardisation

The functions f0—f5 and f1*are proprietary to the home environment. Examples of the functions f1, f1* and f2 are

CBC-MACs or H-MACs [3].

5.1.5 Implementation and operational considerations

The functions f1—f5 and f1* shall be designed so that they can be implemented on an IC card equipped with a 8-bit microprocessor running at 3.25 MHz with 8 kbyte ROM and 300byte RAM and produce AK, XMAC-A, RES, CK and IK in less than 500 ms execution time.

5.1.6 Type of algorithm

5.1.6.1 f0

f0: the random challenge generating function

f0: (internal state) → RAND

f0 should be (pseudo) random number generating function.

5.1.6.2 f1

f1: the network authentication function

f1: (K; SQN, RAND, ~~MODE~~AMF) → MAC-A (or XMAC-A)

f1 should be a MAC function. In particular, it shall be computationally infeasible to derive K from knowledge of RAND, SQN, ~~MODE~~ AMF and MAC-A (or XMAC-A).

5.1.6.3 f1*

f1*: the re-synchronisation message authentication function

f1*: (K; SQN, RAND, ~~MODE~~AMF) → MAC-S (or XMAC-S)

f1 should be a MAC function. In particular, it shall be computationally infeasible to derive K from knowledge of RAND, SQN, ~~MODE~~ AMF and MAC-S (or XMAC-S).

5.1.6.4 f2

f2: the user authentication function

f2: (K; RAND) → RES (or XRES)

f2 should be a MAC function. In particular, it shall be computationally infeasible to derive K from knowledge of RAND and RES (or XRES).

5.1.6.5 f3

f3: the cipher key derivation function

f3: (K; RAND) → CK

f3 should be a key derivation function. In particular, it shall be computationally infeasible to derive K from knowledge of RAND and CK.

5.1.6.6 f4

f4: the integrity key derivation function

f4: (K; RAND) → IK

f4 should be a key derivation function. In particular, it shall be computationally infeasible to derive K from knowledge

of RAND and IK.

5.1.6.7 f5

f5: the anonymity key derivation function

f5: (K; RAND) → AK

f5 should be a key derivation function. In particular, it shall be computationally infeasible to derive K from knowledge of RAND and AK.

The implementation of f5 is optional.

5.1.7 Interface

5.1.7.1 K

K: the ~~user authentication key~~subscriber authentication key

K[0], K[1], …, K[127]

The length of K is 128 bits. The ~~user authentication key~~subscriber authentication key K is a long term secret key stored in the USIM and the AuC.

5.1.7.2 RAND

RAND: the random challenge

RAND[0], RAND[1], …, RAND[127]

The length of RAND is 128 bits.

5.1.7.3 SQN

SQN: the sequence number

SQN[0], SQN[1], …, SQN[~~X10-1~~47]

The length of SQN is ~~between 32 and 64~~48 bits. The AuC should include a fresh sequence number in each authentication token. The verification of the freshness of the sequence number by the USIM constitutes to entity authentication of the network to the user.

~~5.1.7.4 MODE~~5.1.7.4 AMF

~~MODE~~AMF: the ~~mobility management identifier~~authentication management field

~~MODE[0]~~AMF[0], AMF[1], …, AMF[15]

~~The length of MODE is 1 bit. MODE identifies an instance of mobility management, when more than one mobility management is simultaneously active for a single user.~~

The length of AMF is 16 bits. The use of AMF is not standardised. Example uses of the AMF are provided in annex F of TS 33.102.

5.1.7.6 MAC-A (equivalent for XMAC-A)

MAC-A: the message authentication code used for authentication of the network to the user

MAC-A[0], MAC-A[1], …, MAC-A[63]

The length of MAC-A is 64 bits. MAC-A authenticates the data integrity and the data origin of RAND, SQN and ~~MODE~~AMF. The verification of MAC-A by the USIM constitutes to entity authentication of the network to the user.

5.1.7.7 MAC-S (equivalent for XMAC-S)

MAC-S: the message authentication code used to provide data origin authentication for the synchronisation failure information sent by the USIM to the AuC.

MAC-S[0], MAC-S[1], …, MAC-S[63]

The length of MAC-S is 64 bits. MAC-S authenticates the data integrity and the data origin of RAND, SQN and ~~MODE~~AMF. MAC-S is generated by the USIM and verified by the AuC.

5.1.7.8 RES (or XRES)

RES: the user response

RES[0], RES[1], …, RES[~~X21~~31…127]

~~or XRES: the expected user response~~

~~XRES[0], XRES[1], …, XRES[X21]~~

The maximum length of RES and XRES is 128 bits and the minimum is 32 bits. RES and XRES constitute to entity authentication of the user to the network.

5.1.7.9 CK

CK: the cipher key

CK[0], CK[1], …, CK[127]

The length of CK is 128 bits. In case the effective key length should need to be made smaller than 128 bits, the most significant bits of CK shall carry the effective key information, whereas the remaining, least significant bits shall be set zero.

5.1.7.10 IK

IK: the integrity key

IK[0], IK[1], …, IK[127]

The length of IK is 128 bits. In case the effective key length should need to be made smaller than 128 bits, the most significant bits of IK shall carry the effective key information, whereas the remaining, least significant bits shall be set zero.

5.1.7.11 AK

AK: the anonymity key

AK[0], AK[1], …, AK[~~X10-1~~47]

The length of AK is ~~between 32 and 64~~48 bits. It equals the length of SQN.

Annex B:
Functions for alternative AKA protocol

B.1 Scope

The mechanism described here achieves mutual authentication and key agreement between the USER (e.g.,USIM) and the AuC in the user's HE, showing knowledge of a secret key K which is shared between and available only to these two parties. The temporary authentication key generated during execution of the protocol is shared with the visited SN/VLR, and can be used subsequently with the local authentication and session key agreement protocol described in section D.2 of Annex D of [1] or with the other local authentication mechanisms described in Annex D [1]. Additionally, the session keys for the first session are created during execution of the protocol.

Architectural detail can be obtained from Annex D of [1].

B.2 Abbreviations

The following abbreviations are complementing the abbreviations defined above in section 3.3 of this document:

KT Temporary Authentication Key

RES1 Response to the USER-generated Challenge (HE/AuC → SN/VLR → USER)

RES2 Response to the Network-generated Challenge (USER→ SN/VLR → HE/AuC)

RSn1 Network Generated Challenge (HE/AuC → SN/VLR → USER)

Rsu USER-generated Challenge (USER → SN/VLR → HE/AuC)

SN Serving Node

XRES1 Expected Response to the Network-generated Challenge (RES1)

XRES2 Expected Response to the User-generated Challenge (RES2)

B.3 General algorithm requirements

The requirements listed in the body of the document should also apply to the **AKA** protocol. Only the differences will be described in this Annex.

1 RSn and/or RSu can be a random number or a counter

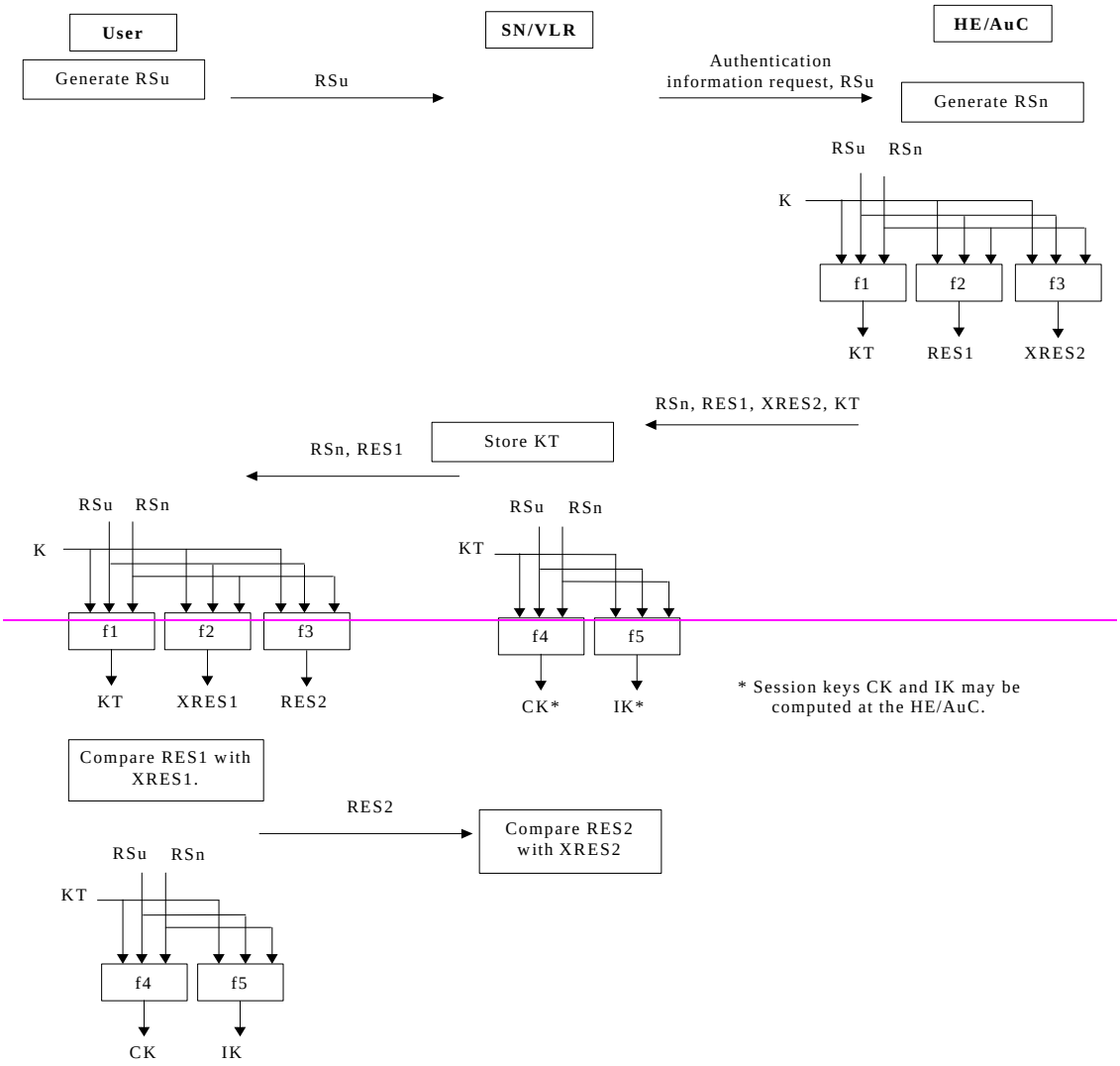
B.4 Functional algorithm requirements

B.4.1 Overview

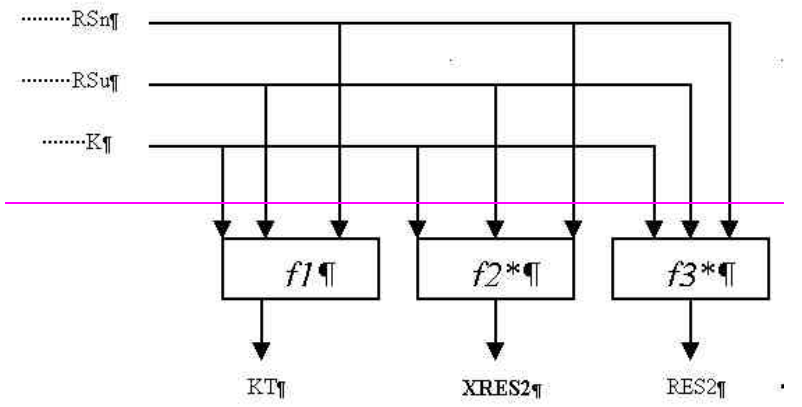
The mechanism for authentication and key agreement described in Annex D of [1] requires the following cryptographic functions:

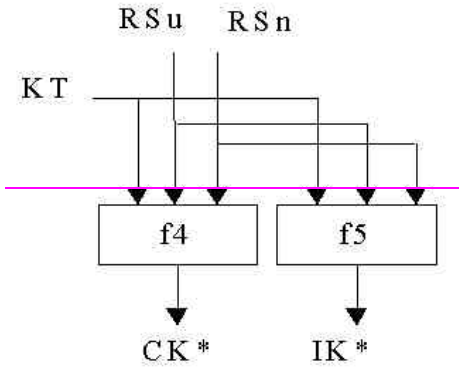
- **f1** Temporary Authentication Key Generation Function (KT)
- **f2** Function to Calculate Response to User Challenge (RES1) or the Expected Response to Network Challenge (XRES1)
- **f3** Function to Calculate the Response to the Network Challenge (RES2) or the Expected Response to User Challenge (XRES2)
- **f4** Function to Derive the Cipher Key (Ck)
- **f5** Function to Derive the Integrity Key (Ik)

Figure B.1 below illustrates the use of the cryptographic functions f1 – f5 at the HE/AuC, SN/VLR and USER.



~~Figure B.1: Functions for generating the Temporary Key, the Ciphering Key, the Integrity Key and the responses to the authentication challenges (X)RES1 and (X)RES2~~





The input parameters to the **f1, f2**, and **f3**(see note) algorithms are:

- the **K** (Authentication Key)

NOTE*: When authentication is performed locally, between the SN/VLR and the User, the **K** input for **f2,f3** is replaced by the **f1** derived Temporary Key **KT** for deriving the **(X)RES2**.

- the **RSu** (Random Number User)
- the **RSn** (Random Number Network)

The input parameters for the **f4**, **f5** algorithms are:

- the Temporary Key (**KT**) generated by **f1**
- the **RSu**
- the **RSn**

B.4.2 Use

The functions **f1 f5** shall be used only to provide mutual entity authentication between the User and the AuC (or the SN/VLR when the temporary authentication key **KT** is shared). The derived keys shall protect the user and the transmission over the radio access link (voice or data). The confidentiality of the user identity should be protected.

B.4.3 Extend of standardisation

The **f2 f5** should be standardise by ETSI SAGE. The **f1** is proprietary to the home environment.

B.4.4 Allocation

The functions **f1 f5** are allocated to the Home Entity / Authentication Centre (AuC) and the USER (e.g., UIM). The **f2 f5** can also be allocated to the SN/VLR where the permanent key (**K**) is replaced, for the currently active

registration period, by the Temporary Key (**KT**) generated at the HE/AuC by **f1**.

B.4.5 Implementation and operational considerations

ATK implementation shall conform to the same requirements as described in section 5.1.5 of the document.

B.4.6 Type of algorithms

B.4.6.1 f0

This function is a pseudo random number generator, as described above in this document. It is used to generate a network random number (RSn) as well as the USER random number (RSu).

f0: (internal state) → RSn

f0: (internal state) → RSu

B.4.6.2 f1

Authentication function that generates the temporary authentication key used, by the serving system, to generate the session ciphering, the integrity keys or perform local authentication. This function is not shared with the SN/VLR and does not have to be standardized.

f1: (K, RSu, RSn) → KT

It should be computationally unfeasible to derive the K from the knowledge of RSu, RSn. or/and KT.

B.4.6.3 f2

Network authentication function deriving the response and the expected response to an authentication challenge from the AuC.

f2: (K, RSn, RSu) → RES1 (in the AuC)

f2: (K, RSn, RSu) → XRES1 (in the USER terminal)

When a local authentication session key agreement is executed, the **KT** is used in the computation instead of the **K** (the SN/VLR is sending the authentication challenge to the USER).

f2: (KT, RSn, RSu) → RES1 (in the SN/VLR)

f2: (KT, RSn, RSu) → XRES1 (in the USER terminal)

It should be computationally unfeasible to derive the K or KT from the knowledge of RSu, RSn. or/and (X)RES1.

B.4.6.4 f3

Network authentication function deriving the response and the expected response to an authentication challenge from the USER terminal.

f3: (K, RSn, RSu) → RES2 (in the USER terminal)

f3: (K, RSn, RSu) → XRES2 (in the AuC)

When a local authentication session key agreement is executed, the **KT** is used in the computation instead of the **K**.

f3: (KT, RSn, RSu) → RES2 (in the USER terminal)

f3: (KT, RSn, RSu) → XRES2 (in the SN/VLR)

It should be computationally unfeasible to derive the K or KT from the knowledge of RSu, RSn. or/and (X)RES2.

B.4.6.5 f4

Function used to generate a session's ciphering key used to encode the over the air conversation between the user's terminal and the serving system.

f4: (KT, RSn, RSu) → Ck (in the USER terminal)

f4: (KT, RSn, RSu) → Ck (in the SN/VLR)

It should be computationally unfeasible to derive the K or KT from the knowledge of RSu, RSn. or/and Ck.

B.4.6.6 f5

Function used to generate a session's integrity key used to encode the over the air signaling between the user's terminal and the serving system.

f5: (KT, RSn, RSu) → Ik (in the USER terminal)

f5: (KT, RSn, RSu) → Ik (in the SN/VLR)

It should be computationally unfeasible to derive the K or KT from the knowledge of RSu, RSn. or/and Ik.

B.4.7 Interface

The following additions are proposed to the interfaces described above in this document.:

B.4.7.1 KT

KT: the user temporary authentication key.

The length of the KT should be the same as specified for K above.

B.4.7.2 RSn and RSu

RSn, RSu: random numbers

The length of RSn and RSu should be the same as specified for the RAND above.

B.4.7.3 RES1, XRES1, RES2, XRES2

RES1 or XRES1: authentication challenge from the network or the expected response from the user.

RES2 or XRES2: authentication challenge from the USER or the expected network response.

The length of those parameters should be the same as specified for the RES above.

Annex C: Unspecified values

Reference	Meaning	Range	Source
X1	Bus width of the USIM processor (bit)		TSG T WG3
X2	Clock speed of the USIM processor (MHz)		TSG T WG3
X3	Memory size of the USIM (kbits)		TSG T WG3
X4	Response time for AK, MAC-A and RES (ms)		TSG SA WG2
X5	Response time for CK and IK (ms)		TSG SA WG2
X6	Bus width of the AuC processor (bit)		TSG CN
X7	Clock speed of the AuC processor (MHz)		TSG CN
X8	Memory size of the AuC (kbits)		TSG CN
X9	Response time for authentication vector in AuC (ms)		TSG SA WG2
X10	Length of sequence number (bits)	32 - 64	TSG SA WG3
X11	Response time for EMUI computation in the USIM (ms)		TSG SA WG2
X12	Response time for SEQ_UIC || IMUI in the AuC (ms)		TSG SA WG2
X13	Length of the group key (bits)	128	TSG SA WG3
X14	Length of SEQ_UIC (bits)	32	TSG SA WG3
X15	Length of IMUI (bits)		TSG SA
X16	Length of EMUI (bits)	128	TSG SA WG3
X17	Number of gates required for hardware implementation of ciphering algorithm	10 000	TSG T WG3 TSG CN
X18	Length of the field LENGTH for ciphering (bits)		TSG RAN WG2
X19	Maximum length of a signalling message (bits)		TSG SA WG3 TSG RAN WG2
X20	Length of MAC-I (bits)	24	TSG SA WG3
X21	Length of RES and XRES (bits)	32-128	TSG SA WG3