

1 3GPP2 S.S0127-0  
2 Version 1.0  
3 Version Date: 19 June, 2008  
4  
5  
6  
7  
8  
9



3RD GENERATION  
PARTNERSHIP  
PROJECT 2  
"3GPP2"

## 10 CAVE Based IMS Security

---

11  
12  
13  
14  
15  
16  
17  
18  
19  
20

### *COPYRIGHT NOTICE*

*3GPP2 and its Organizational Partners claim copyright in this document and individual Organizational Partners may copyright and issue documents or standards publications in individual Organizational Partner's name based on this document. Requests for reproduction of this document should be directed to the 3GPP2 Secretariat at [secretariat@3gpp2.org](mailto:secretariat@3gpp2.org). Requests to reproduce individual Organizational Partner's documents should be directed to that Organizational Partner. See [www.3gpp2.org](http://www.3gpp2.org) for more information.*

21  
22

1           **EDITOR**

2           Zhibi Wang  
3           Alcatel-Lucent  
4           [zhibiwang@alcatel-lucent.com](mailto:zhibiwang@alcatel-lucent.com)

5           **REVISION HISTORY**

---

6

<b>REVISION HISTORY</b>		
<b>1.0</b>	<i>Initial Publication Version</i>	<i>June 2008</i>

7

# Table of Contents

1			
2	1	Introduction.....	1
3	2	Scope.....	1
4	3	References.....	1
5		3.1 Normative References .....	1
6		3.2 Informative References.....	2
7	4	Definitions and Abbreviations .....	2
8		4.1 Definitions .....	2
9		4.2 Abbreviations .....	2
10	5	High Level Principles .....	3
11	6	Requirements .....	5
12		6.1 ME Requirements.....	5
13		6.2 HSS Requirements.....	7
14		6.3 AKA Vector Emulation .....	9
15	7	Call Flows.....	10
16		7.1 IMS Authentication .....	10
17		7.2 Re-Synchronization Procedure .....	12
18		7.3 Get CAVE Credentials From HLR/AC .....	14

# 1 Introduction

---

IP Multimedia Subsystem (IMS) security is defined for cdma2000®<sup>1</sup> networks in [3]. This document [3] defines how the SIP signaling is protected between the User Equipment (UE) and the Proxy Call Session Control Function (P-CSCF), how the subscriber is authenticated and how the subscriber authenticates the IMS using AKA authentication credentials present at the UE.

This document defines the mechanism for secure access to the IMS for UEs equipped with legacy Removable User Identity Modules (R-UIM). The legacy R-UIMs do not support AKA authentication, but only support CAVE authentication based on the A-Key shared between the R-UIM and the HLR/AC.

Furthermore, the IMS HSS does not contain any CAVE authentication information related to the legacy R-UIM and is only available from the HLR/AC.

In this document, several key words are used to signify the requirements. The key words “shall”, “shall not”, “should”, “should not” and “may” are to be interpreted as described in the TIA Engineering Style Manual.

## 2 Scope

---

This document defines the stage-2/3 procedures for the IMS security based on the CAVE authentication.

This document only covers the aspects that differ from the procedures defined in [3] for IMS security based on the CAVE authentication. Unless otherwise specified in this document, the IMS security procedures shall comply with [3].

## 3 References

### 3.1 Normative References

---

- [1] IETF RFC 2617 (1999): "HTTP Authentication: Basic and Digest Access Authentication".
- [2] IETF RFC 3310 (2002): "Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA)".
- [3] 3GPP2 S.S0086: "IMS Security Framework".
- [4] 3GPP2 X.S0004-540-E: "MAP Operations Signaling Protocols".
- [5] 3GPP2 S.S0078: "Common Security Algorithms".
- [6] 3GPP2 S.S0055: "Enhanced Cryptographic Algorithms".
- [7] 3GPP2 C.S0005-D v2.0: "Upper Layer (Layer 3) Signaling Standard for cdma2000 Spread Spectrum Systems, Release D", October 2005.

---

<sup>1</sup> cdma2000® is the trademark for the technical nomenclature for certain specifications and standards of the Organizational Partners (OPs) of 3GPP2. Geographically (and as of the date of publication), cdma2000® is a registered trademark of the Telecommunications Industry Association (TIA-USA) in the United States

## 3.2 Informative References

---

<1> 3GPP2 C.S0023-0: "Removable User Identity Module for Spread Spectrum Systems".

# 4 Definitions and Abbreviations

## 4.1 Definitions

---

For the purposes of the present document, the following terms and definitions apply:

**User Equipment (UE):** For the purposes of this document, the User Equipment is considered as two separate entities, consisting of the User Identity Module (UIM) and the Mobile Equipment (ME). The ME contains a higher power processor.

**Removable UIM (R-UIM):** An UIM that can be physically removed from the UE. The R-UIM can be either a stand-alone module as defined in <1>, or a multi-application platform (also called a UICC) that may hold several applications that can be operated concurrently (e.g. ISIM application, cdma2000 SIM application).

**User Identity Module (UIM):** The User Identity Module is a lower power processor that securely stores, among other things, the security credentials. The User Identity Module may be a Removable UIM (R-UIM) or part of the UE itself.

## 4.2 Abbreviations

---

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

AC	Authentication Center
AKA	Authentication and Key Agreement
AUTN	Authentication Token
CAVE	Cellular Authentication and Voice Encryption
CDMA	Code Division Multiple Access
CK	Cipher Key
ESN	Electronic Serial Number
HLR	Home Location Register

1	HSS	Home Subscriber Server
2	IETF	Internet Engineering Task Force
3	IK	Integrity Key
4	IMS	IP Multimedia Subsystem
5	IMPI	IM Private user Identity
6	IMPU	IM Public User Identity
7	IMSI	International Mobile Station Identity
8	MAC	Message Authentication Code
9	MD5	Message Digest version 5
10	ME	Mobile Equipment
11	NAI	Network Access Identifier
12	P-CSCF	Proxy Call Session Control Function
13	PLCM	Private Long Code Mask
14	RADIUS	Remote Authentication Dial In User Service
15	R-UIM	Removable User Identity Module
16	UIMID	User Identity Module Identifier
17	SHA-256	Secure Hash Algorithm 256
18	SQN	Sequence Number
19	SSD	Shared Secret Data
20	UIM	User Identity Module
21	VLR	Visited Location Register

## 22 **5 High Level Principles**

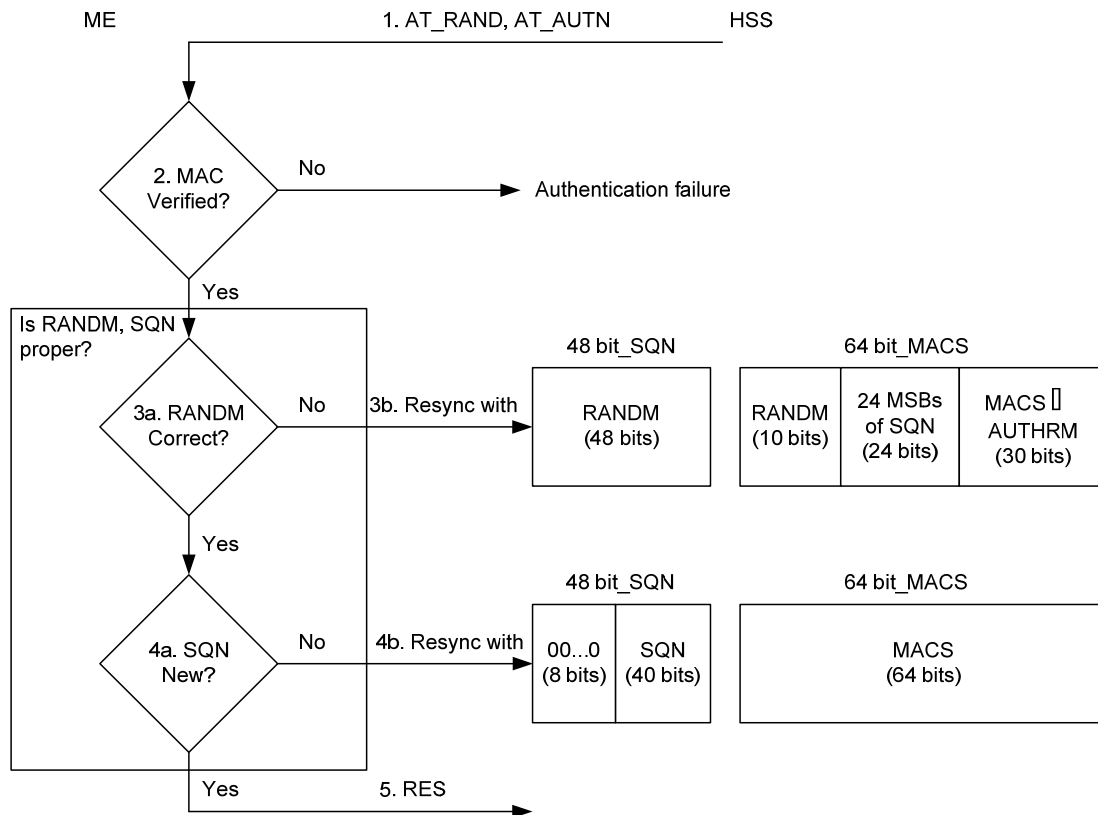
---

23 All procedures closely follow currently defined procedures in [3]. Such as:

- 1 • The same HTTP Digest AKA [1, 2] as specified in [3] is used for mutual authentication and for
- 2 establishing the security association between the UE and the P-CSCF.
  
- 3 • The AKA Authentication Vector is created in the HSS, while the AKA root key for it is
- 4 generated from two set of CAVE KEYS obtained from the HLR, as a result of successful
- 5 CAVE-based authentications.
  
- 6 • The AKA procedures and algorithms are terminated at the ME instead of at the legacy R-UIM.
- 7 The legacy R-UIM is used for performing conventional CAVE-based authentications and
- 8 returning its result to the ME. The ME will use the results of successful CAVE-based
- 9 authentications for performing AKA processing [5, 6].
  
- 10 • Standard AKA functions are used after AKA\_KEY derivation [5, 6].
  
- 11 • The IMPI and IMPU of subscriber are created from the IMSI.
  
- 12 • In all the CAVE calculations [5, 6], ESN is set to 32 LSBs of UIMID.

13 In case CAVE based AKA is used in multiple contexts, the AKA\_KEY and SQN are shared among all  
 14 the contexts.

15 The figure below illustrates the high-level call flow for using CAVE with IMS AKA.



16

17

**Figure 1 High Level flow of CAVE based IMS AKA authentication**

1 Both the ME and the HSS establish a SQN (sequence number) and a RANDM (Mobile's Random Number)  
 2 using the AKA re-synchronization procedure. Once agreed, SQN is used to provide replay protection for the  
 3 AKA authentication. How the AKA fields are used with CAVE is described in more details in the  
 4 subsequent sections.

5 In this document, the MS identifier is assumed to be true IMSI or MIN based IMSI. In case of true IMSI, the  
 6 IMSI\_S1 and IMSI\_S2 should be used in the place of MIN1 and MIN2 [7].

## 7 6 Requirements

### 8 6.1 ME Requirements

---

9 In order to bootstrap the AKA root key using the R-UIM security functions, the ME shall support the  
 10 following requirements:

- 11 • The ME shall be able to translate the authentication interrogation contents received from the HTTP  
 12 Digest AKA messages into the CAVE authentication requests similar to those issued by ME to the  
 13 R-UIM while accessing the cdma2000 systems using CAVE authentication.
- 14 • The ME shall terminate and process the HTTP Digest AKA procedures specified in [3] by using the  
 15 CAVE session keys received from the R-UIM.
- 16 • The ME shall be able to communicate the response to the IMS network, as specified in [3].
- 17 • In case CAVE based AKA is used by the ME in multiple contexts, the ME shall use the same  
 18 AKA\_KEY and SQN for all the contexts.
- 19 • The ME shall use 32 LSBs of the R-UIM's UIMID as its ESN.

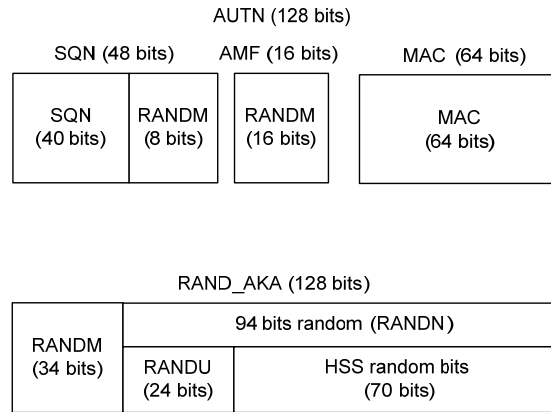
20 Upon detecting the insertion of a new R-UIM, the ME shall generate a 58 bit RANDM value according  
 21 to the following rules:

- 22 • The 8 MSBs (bits 57 - 50) shall be random, but not be all zeroes.
- 23 • The 24 bits following the 8 MSBs shall be random (bits 49-26).
- 24 • The 20 bits following the 32 MSBs (bits 25 - 6) shall be random, but shall have a decimal value  
 25 from 000,000 to 999,999.
- 26 • The remaining 6 LSBs (bits 5 - 0) shall be random.

27 The ME shall store the generated RANDM as  $RANDM_{ME}$ .

28 Upon detecting the insertion of a new R-UIM, the ME also stores a 40 bit SQN value,  $SQN_{ME}$ , which is  
 29 initialized to be a 24 bit value, TIME (bits 39 - 16), followed by 16 zeroes (bits 15 - 0). TIME is set by  
 30 counting the 20 second intervals modulo  $2^{24}$  that have elapsed from the beginning of January 1st 2008  
 31 till present time.  $SQN_{ME}$  is updated during Re-synchronization procedures and is incremented after each  
 32 successful challenge verification.

1 Upon receiving the 401 Auth\_Challenge message [3], the ME shall extract RANDM, RANDN (94  
 2 random bits of RAND\_AKA) and SQN from the received AUTN and RAND as shown below (see 6.3  
 3 for how to construct AUTN and RAND parameters).



4

5

**Figure 2 AKA-Challenge Parameters Format**

6

7

8

If the ME finds that the received RANDM is equal to  $RANDM_{ME}$  and has the  $KEYSM_{ME}$  corresponding to the  $RAND_{ME}$  that was previously calculated and stored at the last re-synchronization procedure then the ME shall use the  $KEYSM_{ME}$ , otherwise, the ME shall calculate  $KEYSM$  as follows:

9

- Use the 32 MSBs of RANDM as a CAVE RAND.

10

11

12

13

14

- If the value of the 20 bits (bits 25 – 6) of the received RANDM, converted to decimal format, is between 000,000 and 999,999 then the ME shall treat it as six dialed digits, and issue the Generate Key/VPM Command to the R-UIM to compute CAVE KEYS with the system access type set to call origination; The ME shall set  $KEYSM = SMEKEY|CDMAPLCM|AUTHR$  using the values returned by the R-UIM.

15

16

17

18

19

- If the value of these 20 bits (bits 25 – 6) of the received RANDM when converted to decimal format is 1,000,000 or greater then the ME shall set the system access type to page response and use the 32 MSBs of RANDM as CAVE RAND and issue a Generate Key/VPM Command to the R-UIM to compute the CAVE KEYS. The ME shall set  $KEYSM = SMEKEY|CDMAPLCM|AUTHR$  using the values returned by the R-UIM.

20

21

22

23

The ME shall concatenate the 24 MSBs of the received RANDU with the 8 LSBs of MIN2 to create a 32 bit CAVE RAND and shall issue the Generate Key/VPM Command to the R-UIM to compute CAVE KEYS and AUTHR. The ME shall set  $KEYSN = SMEKEY|CDMAPLCM|AUTHR$  using the values returned by the R-UIM.

24

25

26

27

28

29

30

The ME shall set the AKA\_KEY to 128 MSBs SHA-256( $KEYSM|KEYSN$ ) output. The ME shall validate the authenticity and freshness of the AKA Challenge by first verifying that the received MAC value equals the calculated XMAC (see 6.3 for placement of MAC value in AUTN). The ME shall use the AKA\_KEY to calculate the XMAC value as the output of f1 function (as defined in [5]) with the 64 MSBs of received AUTN as the SQN and AMF inputs of f1 and the received RAND as the RAND input of f1. If validation of MAC parameter fails, the ME shall format and send the REGISTER message with Failure reason set to AuthenticationFailure.

31

32

33

If the MAC equals XMAC, then the ME shall next verify the freshness of the AKA Challenge by checking that the received RANDM and SQN values are proper. If the received RANDM does not equal  $RANDM_{ME}$ , then the ME shall initiate the Re-Synchronization procedure as follows:

1 For the first re-synchronization procedure after new legacy R-UIM insertion, the ME shall use the initial  
 2 SQN generated at the R-UIM insertion. For the subsequent re-synchronization procedures, the ME shall  
 3 first set the new SQN<sub>ME</sub> to be 24 bits of MSBs of stored SQN<sub>ME</sub> plus 1 followed by 16 bits zeroes and  
 4 then shall format and send the REGISTER message with the AUTS parameter set as follows:

- 5       ▪ the 58 MSBs of AUTS are set to RANDM<sub>ME</sub>,
- 6       ▪ the next 24 bits of AUTS are set to the 24 MSBs of SQN<sub>ME</sub>,
- 7       ▪ and the next 30 bits of AUTS are set to the 30 LSBs of MACS XORed with the 18 bit  
 8 AUTHRM on the least significant side (i.e., XORed with 18 LSBs of MACS). AUTHRM is  
 9 the freshly calculated AUTHR value associated with the RANDM<sub>ME</sub> being sent in the AUTS.

10 Using the 32 MSBs of RANDM<sub>ME</sub> as a CAVE RAND and the next 20 bits of RANDM<sub>ME</sub> as six dialed  
 11 digits, the ME shall set the system access type to call origination and issue the Generate Key/VPM  
 12 Command to the R-UIM to freshly compute the AUTHRM and CAVE KEYS value. The returned  
 13 values are stored as KEYSM<sub>ME</sub> = SMEKEY|CDMAPLCM|AUTHRM. The MACS shall be calculated  
 14 using the f1\* [5] function and AKA\_KEY over the RANDM<sub>ME</sub> and 24 MSBs of SQN<sub>ME</sub>. The  
 15 AKA\_KEY is calculated using the RANDU received in the RAND\_AKA parameter of the last AKA  
 16 Challenge request and the RANDM. The f1\* inputs are set as follows: the RAND input of f1\* is set to  
 17 the received RAND value, 24 MSBs of f1\* SQN input is set to 24 MSBs of SQN<sub>ME</sub>, the next 24 bits of  
 18 f1\* SQN input are set to 24 MSBs of RANDM<sub>ME</sub>, the f1\* AMF input is set to the next 16 MSBs of  
 19 RANDM<sub>ME</sub> and the 18 MSBs of the f1\* RAND input are replaced with the remaining 18 bits of  
 20 RANDM<sub>ME</sub>.

21 If the received RANDM equals RANDM<sub>ME</sub> then the ME shall next verify that the received SQN value  
 22 is in the correct range (see 6.3 for placement of SQN in AUTN). ME shall check if SQN<sub>ME</sub> < SQN <=  
 23 SQN<sub>ME</sub> + Delta, where Delta is a value acceptable to the ME (e.g., Delta is equal to 64).

24 If verification of SQN parameter indicates that the AKA Challenge is a repeat, or SQN requires re-  
 25 synchronization, the ME shall format and send the REGISTER message with the AUTS parameter  
 26 formatted as follows:

- 27       ▪ the 8 MSBs of AUTS are set to zeros (as an indication that it is a SQN resynchronization),
- 28       ▪ the next 40 bit of AUTS are set to SQN<sub>ME</sub>,
- 29       ▪ and the remaining 64 bits of AUTS are set to MACS.

30 The MACS shall be calculated using the f1\* function and AKA\_KEY over SQN<sub>ME</sub>. The f1\* inputs for  
 31 MACS calculation are set as follows: the f1\* RAND input is set to the current received RAND value, 8  
 32 MSBs of f1\* SQN input are set to zeroes and the rest of the f1\* SQN input bits are set to SQN<sub>ME</sub>, and  
 33 the f1\* AMF input is set to zero.

34 Upon successful validation of the AKA Challenge, (i.e. both RANDM and SQN is validated), the ME  
 35 shall update SQN<sub>ME</sub> to the received SQN. The ME shall send the REGISTER including RES as  
 36 specified in [3] to the HSS.

## 37 6.2 HSS Requirements

---

38 In order to bootstrap the AKA root key using the R-UIM security functions, the HSS shall support the  
 39 requirements specified in this section:

40 The HSS shall store a 58 bit value RANDM<sub>HSS</sub> and a 40 bit value SQN<sub>HSS</sub> based on the RANDM and  
 41 the SQN received on the last successful resynchronization from the ME.

1 The HSS shall be provisioned with the subscriber's UIMID as part of the subscriber's IMS profile at the  
2 HSS. The ESN in the IS-41 messages sent to HLR shall be set to 32 LSBs of the subscriber's UIMID  
3 (as identified by the IMSI).

4 Upon receiving the AV request from the S-CSCF, the HSS shall send a unique challenge request, the  
5 IS-41 AUTHREQ [4], to HLR with the mobile identities – IMSI and ESN. The ESN is set to 32 bit  
6 LSBs of UIMID and system access type set to flash request. COUNT is set to zero in all AUTHREQs  
7 sent to the HLR.

8 Upon receiving the IS-41 authreq from the HLR containing the RANDU and AUTHU, the HSS shall  
9 generate the  $RAND = RANDU \parallel (8 \text{ LSBs of MIN2})$ , the  $AUTHR = AUTHU$ , and format a second IS-41  
10 AUTHREQ [4] containing the RAND and AUTHR to the HLR with system access type set to page  
11 response.

12 Once the HSS receives the IS-41 authreq from the HLR containing the SMEKEY and CDMAPLCM,  
13 the HSS shall set  $KEYSN = SMEKEY \parallel CDMAPLCM \parallel AUTHR$ .

14 The HSS shall then set the AKA\_KEY to 128 MSBs of the  $SHA-256(KEYSM_{HSS} \parallel KEYSN)$  output from  
15 the stored  $KEYSM_{HSS}$ , associated with  $RANDM_{HSS}$ , and the newly calculated KEYSN.

16 The HSS shall create a 94 bit RANDN by setting the 24 MSBs to RANDU and rest of RANDN to a  
17 random value.  $SQN_{HSS}$  is incremented;  $SQN_{HSS}$ ,  $RANDM_{HSS}$  and RANDN are included in AUTN and  
18 RAND as specified in 6.3.

19 If the HSS has no stored  $RANDM_{HSS}$  for this user (e.g. for the first authentication), the HSS shall use  
20 the above RANDU part of RANDN to create  $RANDM_{HSS}$  and use KEYSN as KEYSM in forming the  
21 AKA\_KEY. In this initial case, the HSS shall set the 32 MSBs of  $RANDM_{HSS}$  as 24 bits of RANDU  
22 and 8 bits of MIN2. The next 20 bits of  $RANDM_{HSS}$  shall be set to the binary representation of the  
23 (decimal) value 1,000,000. The remaining 6 bits of  $RANDM_{HSS}$  are set randomly. Also if the HSS has  
24 no stored  $SQN_{HSS}$  value then the HSS shall initialize it to 0.

25 The HSS shall use the AKA\_KEY to calculate the MAC value as the output of f1 function with the 64  
26 MSBs of created AUTN as the SQN and AMF inputs and the created RAND as the RAND input of f1.  
27 The 64 LSBs of AUTN shall be set to the calculated MAC value as specified in 6.3. The HSS shall  
28 calculate the 128 bit XRES, CK and IK using AKA functions f2, f3 and f4 respectively [6], using the  
29 AKA\_KEY and the created RAND as input.

30 The HSS shall send one AV to the S-CSCF. Note that this HSS behavior is different from the  
31 procedures in [3] where the HSS may send n AVs. The S-CSCF sends 401 Auth\_Challenge with  
32 RAND, AUTN to AT from the received AV.

33 Upon receiving the AUTS from the ME, the HSS shall check to see if the 8 MSBs of AUTS are all  
34 zeros. If they are all zeros then the 40 bits after the 8 zeroes of AUTS are treated as the received SQN  
35 value and the HSS shall create and verify the 64 bit MACS using the AKA\_KEY created at the last  
36 challenge. The MACS shall be calculated using the f1\* function and AKA\_KEY over the received  
37 SQN. The f1\* inputs for MACS calculation are set as follows: the f1\* RAND input is set to the RAND  
38 value sent in the last challenge, 8 MSBs of f1\* SQN input are set to zeroes and the rest of the f1\* SQN  
39 input bits are set to SQN, and the f1\* AMF input is set to zero. If the validation of the 64 bit MACS is  
40 successful, the HSS shall update the  $SQN_{HSS}$  with the received SQN.

41 If the 8 MSBs of AUTS are not all zeros, then the 58 MSBs of AUTS are treated as the received  
42  $RANDM$  and the next 24 bits are treated as the 24 MSBs of SQN. The HSS shall create and verify the

1 30 bit MACS using the AKA\_KEY created for the last challenge. The MACS shall be calculated using  
 2 the  $f1^*$  function and AKA\_KEY over the received RANDM and 24 MSBs of SQN. The  $f1^*$  inputs are  
 3 set as follows: the RAND input of  $f1^*$  is set to the RAND value sent in the last challenge, the 24 MSBs  
 4 of  $f1^*$  SQN input are set to 24 MSBs of SQN received as part of AUTS, the next 24 bits of  $f1^*$  SQN  
 5 input are set to 24 MSBs of the received RANDM, the  $f1^*$  AMF input is set to the next 16 MSBs of  
 6 RANDM and the 18 MSBs of the  $f1^*$  RAND input are replaced with the remaining 18 bits of RANDM.

7 In order to validate the received AUTS, first the 30 LSBs of the calculated MACS shall be XORed with  
 8 the 30 LSBs of AUTS. If the resulting string's 12 MSBs are not all zeroes then the validation has failed.  
 9 Assuming they are all zeroes, the lower 18 bits are treated as the received AUTHRM value. Using the  
 10 32 MSB of RANDM as a CAVE RAND and the next 20 bits of RANDM as six dialed digits, the HSS  
 11 shall set the system access type to call origination and issue an IS-41 AUTHREQ to HLR including the  
 12 CAVE RAND, dialed digits and AUTHRM as parameters. If the HLR responds with success and  
 13 CAVE KEYS then the AUTS is validated and the HSS shall set  $RANDM_{HSS}$  to the received RANDM  
 14 value and set  $KEYSM_{HSS} = SMEKEY|CDMAPLCM|AUTHRM$  values, and also set  $SQN_{HSS}$  equal to  
 15 the received 24 MSBs of SQN concatenated with 16 zero bits. The HSS should store  $KEYSM_{HSS}$  for  
 16 subsequent use.

17 In case CAVE based AKA is used by the HSS in multiple contexts for a given UE, the HSS shall use  
 18 the same AKA\_KEY and SQN for all the contexts.

19

## 20 6.3 AKA Vector Emulation

---

21 This section describes format of the AT\_AUTN and AT\_RAND when the challenge is created and sent  
 22 by the HSS and is received and verified by the ME:

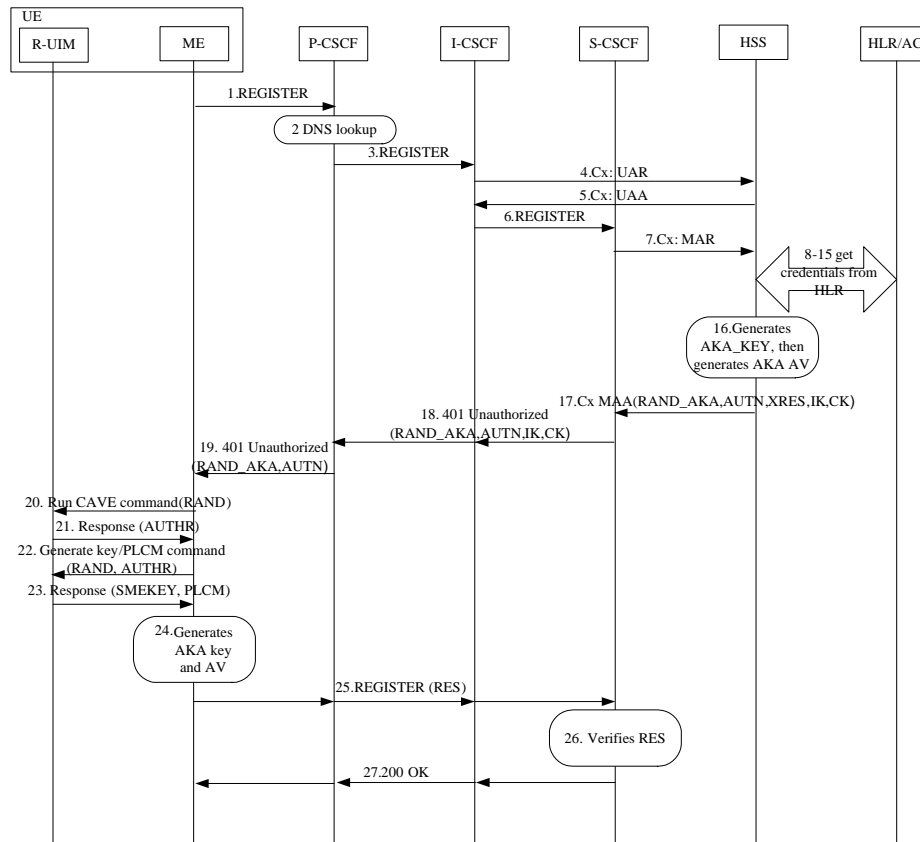
- 23     ▪ 40 MSBs of AUTN are set to SQN,
- 24     ▪ Next 24 MSBs of AUTN are set to 24 MSBs of RANDM,
- 25     ▪ Next 64 bits of AUTN are set to MAC,
- 26     ▪ 34 MSBs of RAND are set to the remaining 34 bits of RANDM,
- 27     ▪ The remaining 94 bits of RAND are set to RANDN.

28

1

2 **7 Call Flows**

3

4 **7.1 IMS Authentication**

5

6 **Figure 3 IMS Authentication Using CAVE**

7

8 1-7. Steps 1-7 are the same as in a normal IMS/MMD message flow.

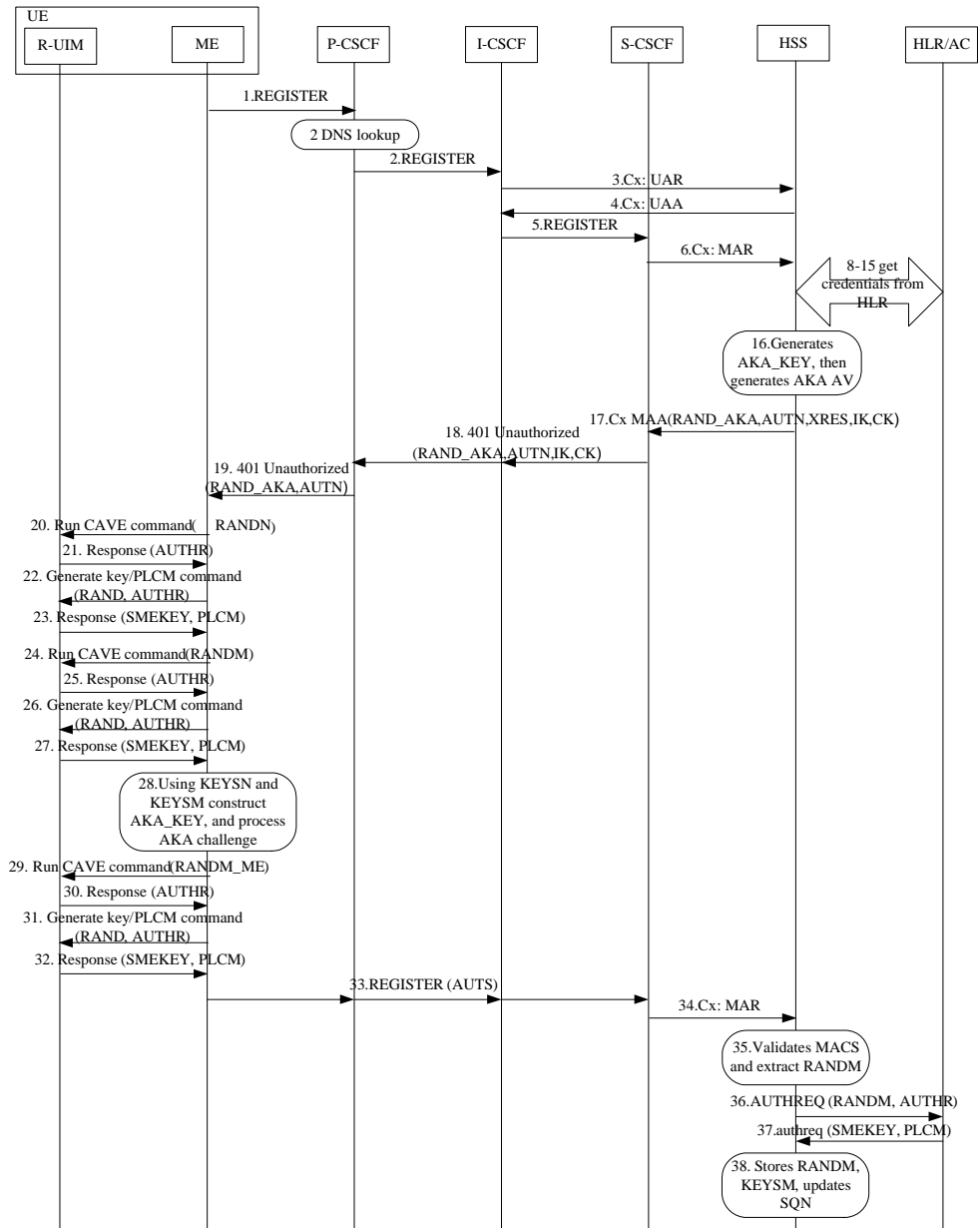
9 8-15. In step 8, when the HSS receives an authentication request from the S-CSCF with the IMPI  
 10 and IMPU, the HSS, based on profile, recognizes that the UE does not support full IMS authentication.  
 11 Instead the UE supports a CAVE-based IMS Authentication. The HSS derives the IMSI from the IMPI  
 12 and IMPU and sends the IS-41 AUTHREQ with the IMSI to the HLR/AC.

13 Note: It is assumed that the IMSI is contained in IMPI as well as IMPU as indicated in X.P0013-002  
 14 section 4.3.3.1.

- 1 See Figure 5 for a detailed call flow to get CAVE keys from HLR/AC. The HSS shall set KEYSN =  
 2 SMEKEY|CDMAPLPCM|AUTHR. The HSS computes the AKA\_KEY from KEYSN and stored  
 3 KEYSM associated with RANDM<sub>HSS</sub>, AKA\_KEY = 128 MSBs of SHA-256(KEYSM|KEYSN).
- 4 16. HSS creates a 94 bit RANDN by setting the 24 MSBs to RANDU and rest of RANDN to a random  
 5 value. SQN<sub>HSS</sub> is incremented; SQN<sub>HSS</sub>, RANDM<sub>HSS</sub> and RANDN are placed in AUTN and RAND as  
 6 specified in 6.3.16. The HSS uses the AKA\_KEY to calculate the MAC value as the output of f1  
 7 function with the 64 MSBs of created AUTN as the SQN and AMF inputs and the created RAND as the  
 8 RAND input of f1. The 64 LSBs of AUTN are set to the calculated MAC value as specified in 6.3. The  
 9 HSS calculates 128 bit XRES, CK and IK using AKA functions f2, f3 and f4 respectively, the  
 10 AKA\_KEY and the created RAND as input. Then the HSS creates the AKA Authentication Vector  
 11 (AV) including AUTN, XRES, IK, CK, RAND\_AKA. The HSS sends MAA back to S-CSCF.
- 12 Steps 17-19 are the same as the normal IMS message flow [3].
- 13 20. From the received challenge in step 19, the ME shall extract RANDM, RANDN and SQN from the  
 14 received AUTN and RAND (see 6.3 for vector format). The ME sends CAVE RAND composed of  
 15 MIN2 and RANDU along with ESN that is set to 32 LSBs of the UIMID to the R-UIM.
- 16 21. The R-UIM responds with AUTHR
- 17 22. The ME instructs R-UIM to generate CAVE keys.
- 18 23. The R-UIM responds to ME with the CAVE KEYS – PLPCM and SMEKEY.
- 19 24. Using the CAVE KEYS returned by R-UIM, the ME creates KEYSN; AKA\_KEY is set to 128  
 20 MSBs of SHA-256 (KEYSM|KEYSN) output. If RANDM is the same as RANDM<sub>ME</sub> stored by the ME  
 21 then the stored KEYSM<sub>ME</sub> is used as KEYSM. Otherwise, the KEYSM is generated. The ME calculates  
 22 XMAC using AKA\_KEY, SQN and RAND\_AKA as input. If the MAC equals XMAC then the ME  
 23 next verifies that the freshness of the AKA Challenge by checking that the received RANDM and SQN  
 24 values are proper. If both RANDM and SQN are in correct range then, the ME updates SQN to the  
 25 received SQN and runs the AKA algorithms to calculate the AKA result (RES, IK, CK) with  
 26 AKA\_KEY and the received RAND, where the RES is 128 bits long.
- 27 The ME uses CK/IK to establish an IPsec SA with the P-CSCF as specified in [3].

1 Steps 25-27 are the same as in normal IMS message flow [3].

## 2 7.2 Re-Synchronization Procedure



3

4

**Figure 4 Re-synchronization Procedure**

- 1 1-19. Steps 1-19 are the same as in section 7.1.
- 2 20. From the received challenge in step 19, the ME extracts RANDM, RANDN and SQN from the  
3 received AUTN and RAND (see 6.3 for vector format). The ME sends CAVE RAND composed of  
4 MIN2 and RANDU from RANDN (see Fig. 2) along with ESN that is set to 32 LSBs of the UIMID to  
5 the R-UIM.
- 6 21. The R-UIM responds with AUTHR.
- 7 22. The ME instructs the R-UIM to generate CAVE keys.
- 8 23. The R-UIM responds to ME with the CAVE KEYS – PLCM and SMEKEY Using the CAVE  
9 KEYS returned by R-UIM, the ME creates KEYSN.
- 10 24. Since the received RANDM doesn't equal  $RANDM_{ME}$ , the ME runs the RUN CAVE command  
11 using received RANDM as RAND.
- 12 25. The R-UIM responds with AUTHR.
- 13 26. The ME instructs R-UIM to generate CAVE keys.
- 14 27. The R-UIM responds to ME with the CAVE KEYS – CDMAPLCM and SMEKEY. Using the  
15 CAVE KEYS returned by R-UIM, the ME creates the KEYSM;
- 16 28. The ME computes AKA\_KEY from KEYSN and KEYSM obtained in steps 23 and 27. The ME  
17 processes RAND and AUTN in the received challenge using the AKA\_KEY. First the ME verifies the  
18 MAC part of AUTN.
- 19 29. Since the received RANDM is different from the stored  $RANDM_{ME}$ , re-synchronization procedure  
20 needs to be performed. The UE generates a fresh  $KEYSM_{ME}$  in the steps before the AUTS is calculated  
21 and sent. The ME sends RAND based on  $RANDM_{ME}$  and dialed digits to the R-UIM.
- 22 30. The R-UIM responds with AUTHRM
- 23 31. The ME instructs R-UIM to generated CAVE keys.
- 24 32. The R-UIM responds to the ME with the CDMAPLCM and SMEKEY. The ME sets and stores  
25  $KEYSM_{ME} = SMEKEY|CDMAPLCM|AUTHRM$ .
- 26 33. The ME issues Register message with failure type set to Synchronization Failure, where AUTS  
27 includes  $RANDM_{ME}$ , 24 MSBs of SQN, AUTHRM and MACS. MACS is computed using the  
28 AKA\_KEY generated in step 28.
- 29 34. The S-CSCF sends the Re-synchronization request to HSS.
- 30 35. The HSS first validates MACS using the using the AKA\_KEY created for the last challenge and  
31 extracts AUTHRM.

1 36. The HSS forms the RAND and dialed digits based on the received RANDM and sends them to the  
 2 HLR along with the received AUTHRM, IMSI and ESN that is set to 32 LSBs of the UIMID in the  
 3 AUTHREQ message.

4 37. The HLR validates the AUTHRM, generates the CDMAPLCM and SMEKEY, and returns them to  
 5 the HSS in the authreq.  $KEYSM_{HSS}$  is set as  $SMEKEY|CDMAPLCM|AUTHRM$ .

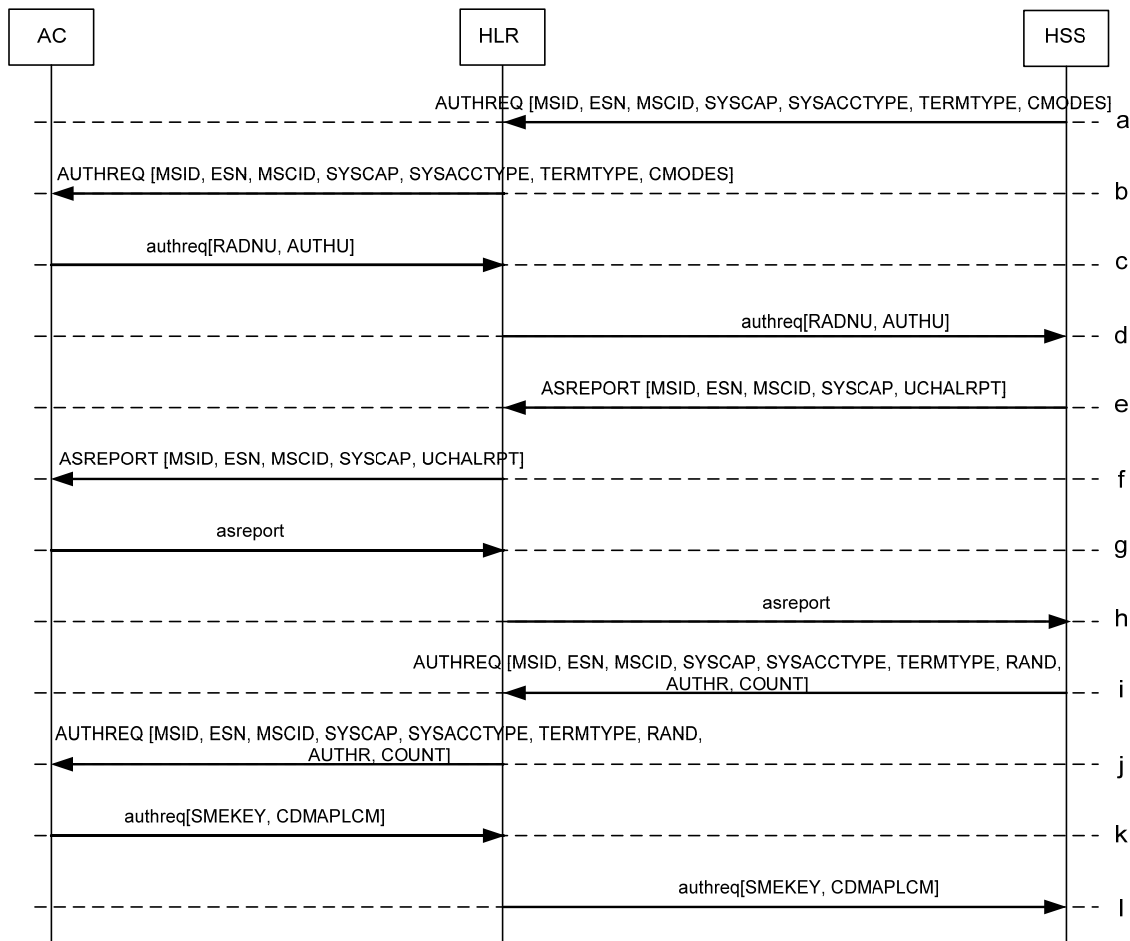
6 38. The HSS stores  $KEYSM_{HSS}$  and the received RANDM as  $RANDM_{HSS}$  for later use, and sets SQN  
 7 equal to the received 24 MSBs of SQN concatenated with 16 zeroes.

8 The rest of the messages are the same as messages starting from Step 17 in Figure 3 of Section 7.1.

9

### 10 7.3 Get CAVE Credentials From HLR/AC

11 This section describes the detailed procedure for HSS interaction with HLR/AC to get the CAVE  
 12 credentials from HLR/AC using command specified in [4] as follows:



13

14

**Figure 5 HSS interactions with HLR/AC**

1 a. An MS accesses a system that is using AKA with CAVE translation for authentication. The  
 2 HSS functions as a VLR for it's interaction with the MS's HLR and AC and sends an AUTHREQ to the  
 3 MS's HLR. The SYSCAP parameter is set to indicate:

4 1. Authentication parameters were not requested on this system access (AUTH=0 in the  
 5 OMT)

6 2. Signaling Message Encryption is supported by the system.

7 3. Voice Privacy is supported by the system.

8 4. System cannot execute the CAVE algorithm and cannot share SSD for the indicated  
 9 MS.

10 5. SSD is not shared with the system for the indicated MS.

11 The TERMTYPE parameter is set to indicate IS-2000 or later (e.g., IS-2000-D). The SYSACCTYPE  
 12 parameter is set to indicate Flash request. The Signaling Message Encryption (SE) Confidentiality  
 13 Status field of the CMODES parameter is set to indicate Off.

14 b. The HLR verifies that the indicated MS is authorized for service and forwards the AUTHREQ  
 15 to the AC.

16 c. In response to the AUTHREQ with these specific set of parameter values, the AC chooses a  
 17 Unique Random Variable (RANDU) and executes CAVE using the SSD-A currently stored, MIN1 and  
 18 MIN2 associated with the MS to produce a Unique Authentication Response (AUTHU). The AC sends  
 19 an authreq to the HLR including RANDU and the expected AUTHU result.

20 Note: This scenario assumes that the AC does not initiate an SSD Update in the response to the serving  
 21 system.

22 d. The HLR forwards the authreq to the HSS.

23 e. The HSS sends an ASREPORT to the HLR with the UCHALRPT parameter set to indicate  
 24 Unique Challenge successful.

25 f. The HLR forwards the ASREPORT to the AC.

26 g. The AC sends an asreport to the HLR.

27 h. The HLR sends the asreport to the HSS.

28 i. The HSS uses the RANDU and AUTHU parameter values received from the AC to produce a  
 29 RAND parameter value and an AUTHR parameter value for a Page response access or call origination.

30 The HSS sends an AUTHREQ to the HLR and includes the generated RAND and AUTHR values. The  
 31 SYSCAP parameter is set to indicate:

- 1           1.       Authentication parameters were requested on this system access (AUTH=1 in the  
2           OMT).
- 3           2.       Signaling Message Encryption is supported by the system.
- 4           3.       Voice Privacy is supported by the system.
- 5           4.       System cannot execute the CAVE algorithm and cannot share SSD for the indicated  
6           MS.
- 7           5.       SSD is not shared with the system for the indicated MS.

8           The TERMTYPE parameter is set to indicate IS-2000 or later (e.g., IS-2000-D). The SYSACCTYPE  
9           parameter is set to indicate Page response. The COUNT parameter is set to a default value.

10          Note: This scenario assumes that the CallHistoryCount was not received from the MS during the initial  
11          MS access and sets the CallHistoryCount parameter to a default value.

12          j.       The HLR forwards the AUTHREQ to the AC.

13          k.       The AC executes CAVE using the value of the RandomVariable (RAND) parameter, the MS's  
14          SharedSecretData (SSD) recorded in the HLR's database. The CAVE authentication result and the  
15          AuthenticationResponse (AUTHR) received from the MS match and the AC includes the SMEKEY and  
16          CDMAPLCM in the authreq sent to the HLR.

17          Note: This scenario assumes that the CallHistoryCount is not verified by the AC and that the current  
18          CallHistoryCount value is not retrieved from an "old" VLR. The scenario also assumes that the AC  
19          does not attempt to initiate an SSD update.

20          l.       The HLR forwards the authreq to the HSS. The MS's profile indicates the Voice Privacy is  
21          authorized and the HLR includes the CDMAPLCM parameter received from the AC.

22          The HSS uses the received SMEKEY and CDMAPLCM parameter values for AKA authentication.