S3-030040

MBMS Security Framework

Phil Hawkes (Australia)
Ramachandran Subramanian (USA)
QUALCOMM
{phawkes,rsubrama}@qualcomm.com

20-Feb-03

Outline

- Security Goals
- MBMS Architecture
- MBMS Functional Architecture
- MBMS Key Hierarchy
- Security Mechanisms
  - Key Management
  - Encryption layer
  - BAK Update

Security Goals

- The problem
  - UICC is not powerful enough to decrypt so ME must decrypt. Decryption keys must be stored in the ME
  - ME is not secure storage. Must assume an attacker may extract the current decryption key from the ME
  - An attacker who is a subscribed user will be able to distribute the decryption key to other non-subscribed users
- In summary:
  - The need to store decryption keys in insecure memory makes it impossible to design a scheme where non-subscribed users CANNOT access the data

Security Goals

- **The goal of the security:**

  Dissuade our potential market from using illegitimate means to access the content

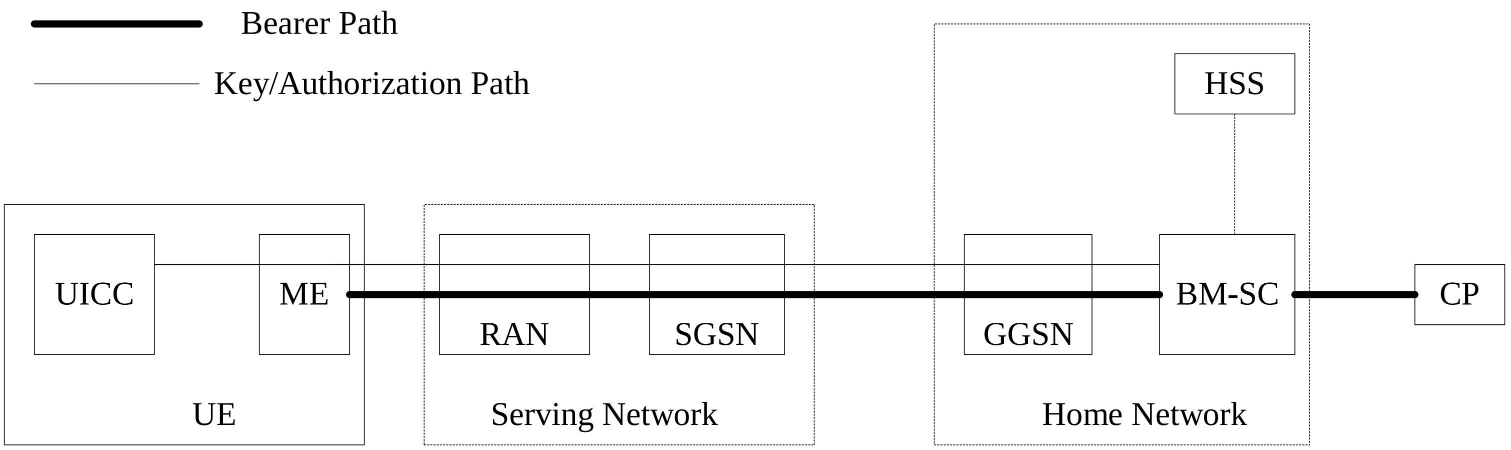
- **What is the potential market?**
  - **Users that desire cheap access to multicast services while being mobile.**

- Attacks we should not be concerned about:
  - Attacks that are expensive to mount (per-user basis) and/or
  - Attacks that assume the user is not mobile.
  - E.g., we should not be concerned about attacks that require the user to perform a frequent data download of keys as mobile data downloads will be more expensive than MBMS subscriptions.

Comment

- This mechanism does not address integrity protection:
  - The nature of single-channel multicast means that small transmission errors will likely occur. Integrity protection would force entire blocks to be discarded due to single bit transmission errors
  - Integrity protection is probably not useful for MBMS

MBMS Architecture



- **Broadcast-multicast service center (BM-SC)**
  - **Schedules multicast data**
  - **Expected to perform most MBMS security functions on network side**
- **Content provider:** **A third party content source**

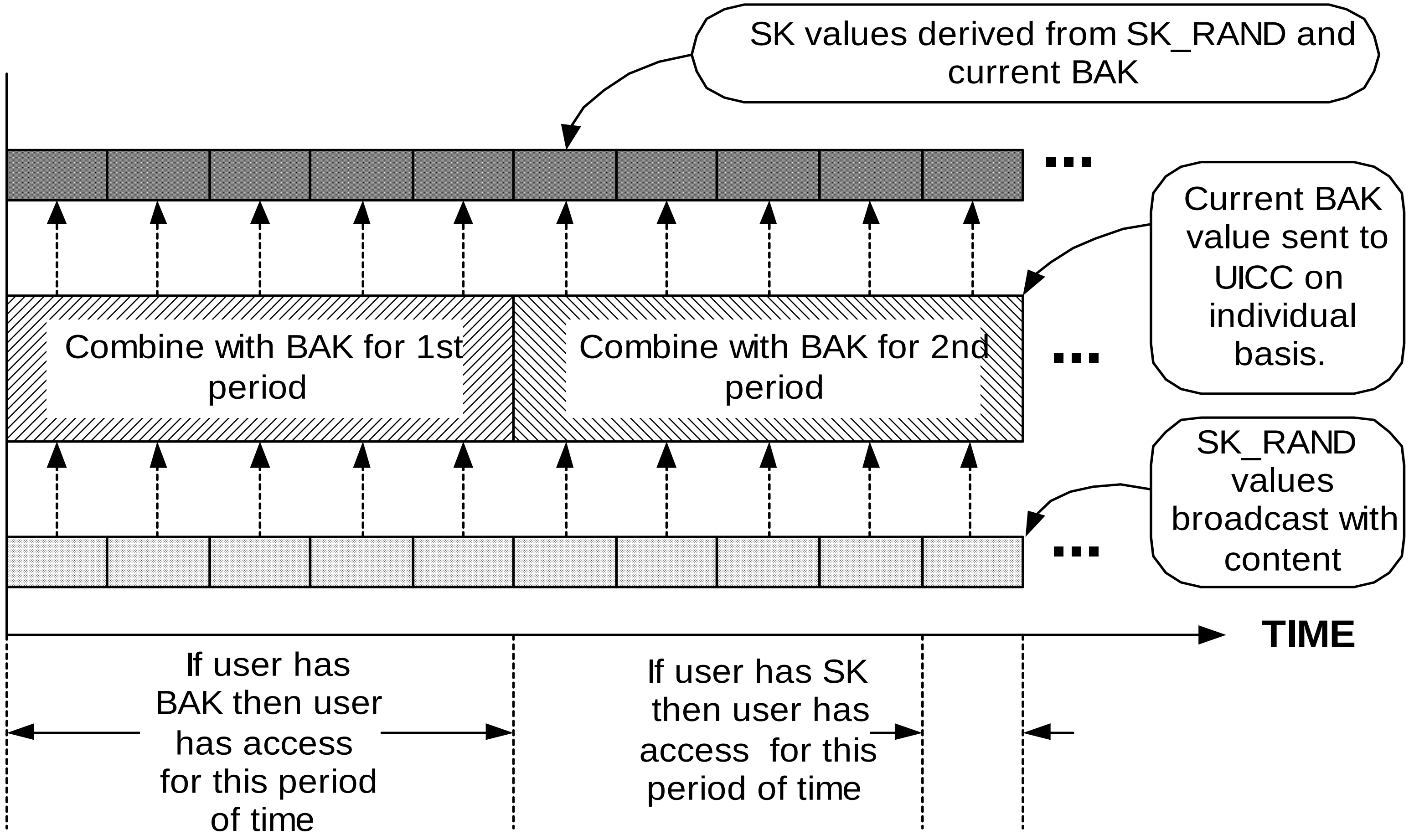
Architecture Entities

- Home Network
  - HSS:
    - holds mobile phone subscription
    - collect billing data for multicast services
  - Owns BM-SC that the user is getting content from
- Serving Network
  - A network that is transmitting the multicast data

MBMS Key Hierarchy

- **RK (Registration Key)** – permanent, user-specific key
  - **Used to generate TK values / authenticate UICC**
- **TK (Temporary Key)** – single use, user-specific key
  - **Used to encrypt BAK values, used by UICC to decrypt BAK values**
  - **Generated using RK**
- **BAK (Broadcast Access Key)** -medium term, shared key
  - **Multiple short-term keys (SK) derived from single BAK**
  - **Distributed to UICC of subscribed users on a per-user basis**
- **SK (Short-term Key)** – frequently changing, shared key
  - ***used to encrypt / decrypt content***
  - **Generated using SK_RAND which is sent in the clear with the encrypted content and BAK**
  - **UICC re-generates SK from BAK and SK_RAND, passes SK to ME**
  - **Changes frequently**

Figure: BAK and SK


SK values derived from SK_RAND and current BAK
...
Current BAK value sent to UICC on individual basis.
Combine with BAK for 1st period
Combine with BAK for 2nd period
...
SK_RAND values broadcast with content
...
TIME
If user has BAK then user has access for this period of time
If user has SK then user has access for this period of time

Using BAK and SK

- To encrypt (in network)
  - SK is generated from BAK and SK_RAND
  - (Multiple SK values generated from each BAK)
  - Content is encrypted with SK
  - Encrypted content is then transmitted along with SK_RAND
- To decrypt (in UE)
  - BAK sent to the UICC of authorized users on a per-user basis
  - ME receives encrypted content and SK_RAND
  - UICC generates SK from BAK and SK_RAND, and passes SK to the ME
  - ME decrypts content using SK

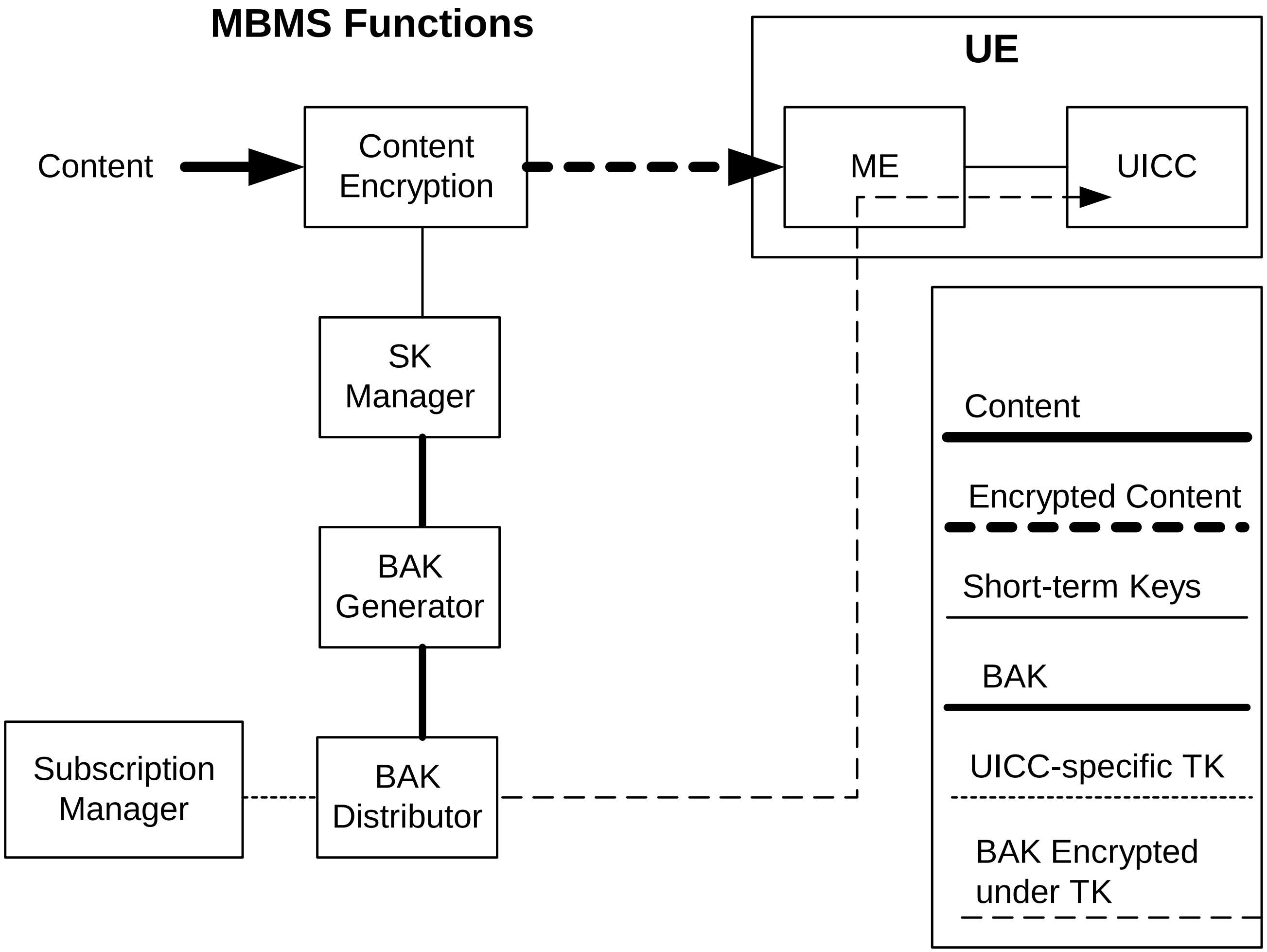
BAK and Subscriptions

- Once the user has a BAK value, the user can access ALL content that is encrypted while BAK value is being used to generate SK values
- Subscriptions grant users the right to be sent BAK values
  - Flat-rate subscriptions: the user is sent all BAK values over a certain time period
  - Event-based subscriptions: the user is sent all BAK values required to view a specific event
  - Usage-based subscriptions: the user is billed based on the number of BAK values used
- The time between changing BAK is a billing decision.
  - Time between BAK changes may vary for each multicast service
  - For a particular multicast service, the time between BAK changes need not stay constant

SK

- ME knows SK
  - Assume subscriber may distribute SK to other users so they can get free access to content
- By changing SK frequently, we limit the amount of time/data that SK is useful to other users
  - To access content, non-subscribed users must download SK values frequently. As discussed in security goals, this attack is not of concern
- The frequency of SK changes may vary for each multicast service and may vary with time
  - When determining how often to change SK, ensure that the cost of user downloading SK exceeds the value of content encrypted under SK

MBMS Functional Architecture



QUALCOMM

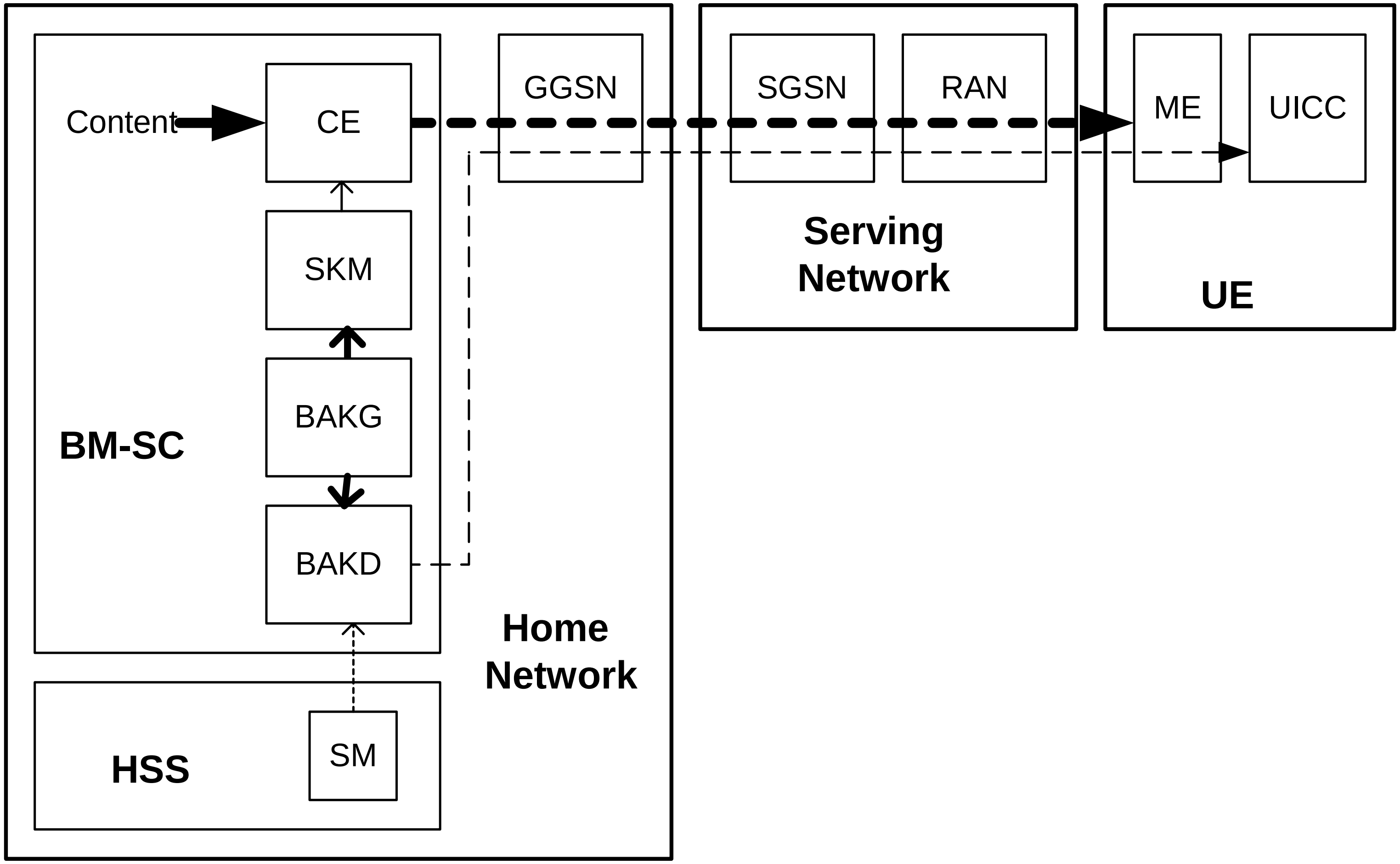
Functional Entities

- MBMS Functions
  - Subscription Manager (SM) holds subscription data authorizing user to some multicast services
  - BAK Generator: generates BAK: forwards to BAKD and SKM
  - BAK Distributor: encrypts BAK for provisioning into UICC
  - SK Manager (SKM): generates SK values from BAK and forwards to Content Encryption (CE).
  - Content Encryption (CE):
- UE
  - Mobile Equipment (ME): performs content decryption
  - UICC: performs key management

Keys and Functional Entities

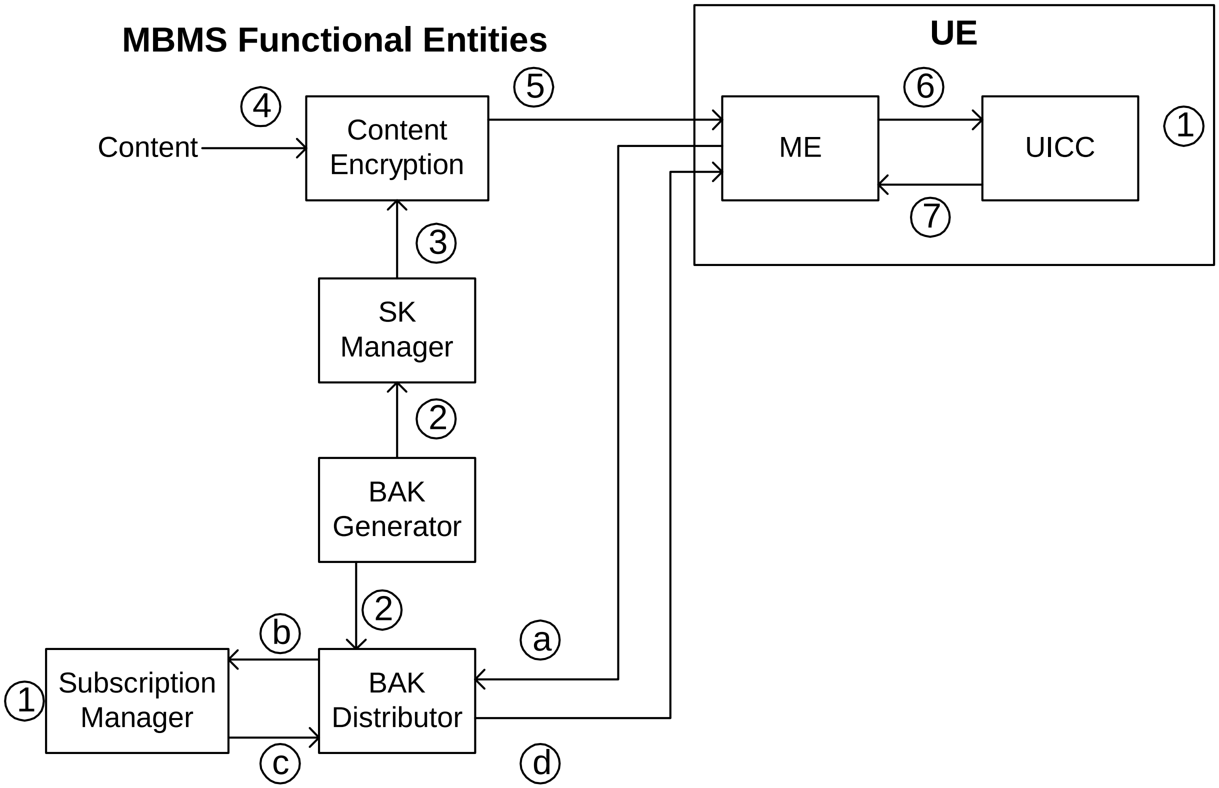
- RK (Registration Key)
  - Used by SM to generate TK values / authenticate UICC
  - Held in Subscription Manager (SM) and UICC
- TK (Temporary Key)
  - Used by BAKD to encrypt BAK values, used by UICC to decrypt BAK values
  - Generated by SM using RK
- BAK (Broadcast Access Key)
  - Multiple decryption keys (SK) derived from single BAK
  - Generated by BAKG and forwarded to BAKD and SKM
  - Distributed by BAKD to UICC of subscribed users (on request)
- SK (Short-term Key)
  - Generated by SKM using BAK and forwarded to CE for encrypting content
  - Derived in UICC using BAK and passed to ME for decrypting content
  - Changes frequently

Expected Mapping of Functions onto Architectural Entities



Security Mechanisms

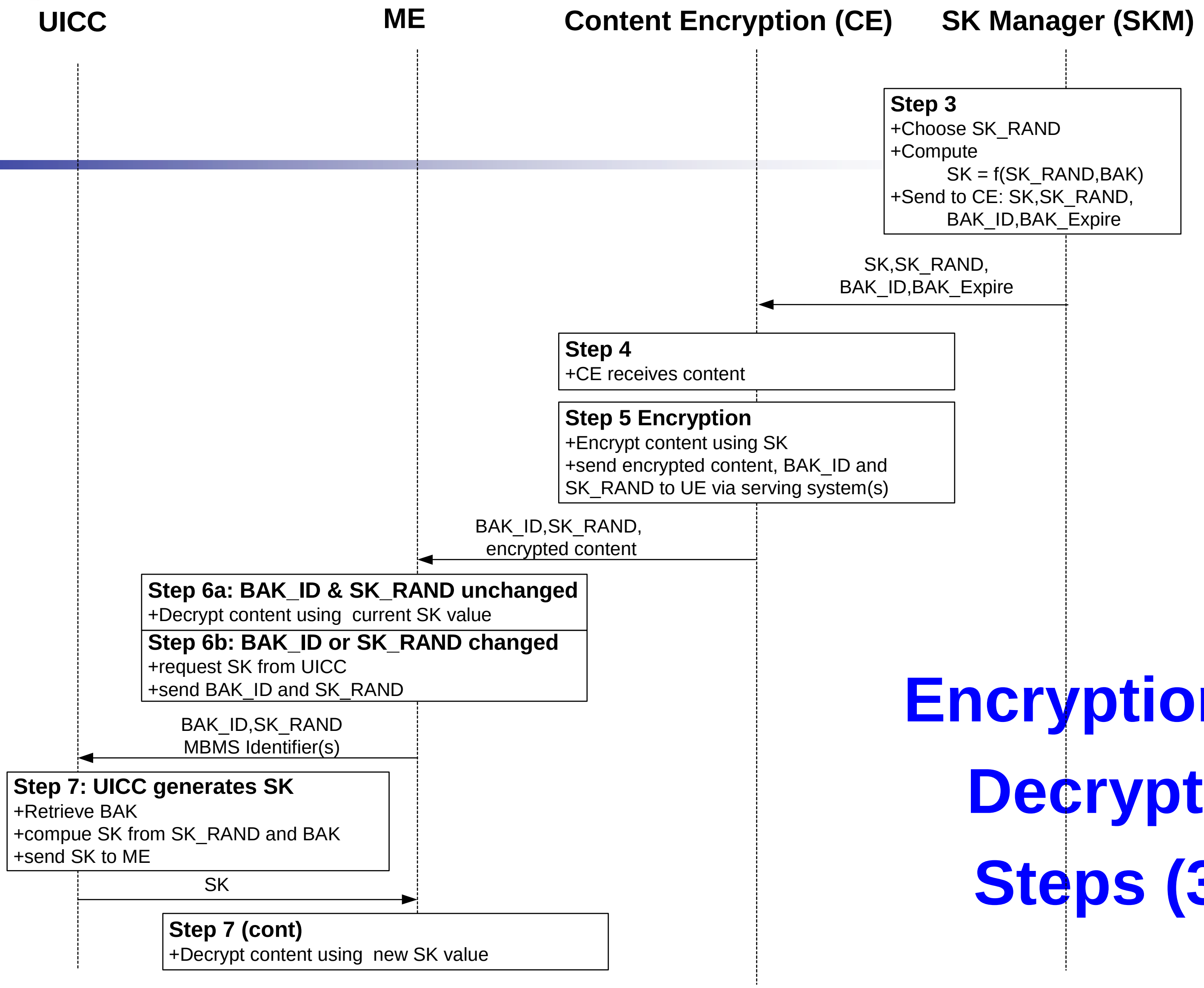
- **Steps (1-2): Prior to ME decrypting Content**
- **Steps (3-7): Encrypting Content**
- **Steps (a-d): BAK Request**



Steps (1-2)

- Step 1. UICC and SM provisioned with RK
  - Beyond scope of this document
    - Could be pre-provisioned in UICC before given to user
    - Could be provisioned OTA
  - This step only needs to occur once to associate SM and UICC
- Step 2. The BAKG generates a value for BAK
  - Associates the value with an identifier BAK_ID and an expiry time (BAK_Expire).
  - The value of BAK, along with the corresponding values of BAK_ID and BAK_Expire, are passed to the SKM and BAKD
  - This step occurs only when the BAKG decides that a new value of BAK should be used.

Encryption & Decryption Steps (3-7)



UICC

ME

Content Encryption (CE)

SK Manager (SKM)

Step 3
+Choose SK_RAND
+Compute
SK = f(SK_RAND,BAK)
+Send to CE: SK,SK_RAND,
BAK_ID,BAK_Expire

SK,SK_RAND,
BAK_ID,BAK_Expire

Step 4
+CE receives content

Step 5 Encryption
+Encrypt content using SK
+send encrypted content, BAK_ID and SK_RAND to UE via serving system(s)

BAK_ID,SK_RAND,
encrypted content

Step 6a: BAK_ID & SK_RAND unchanged
+Decrypt content using current SK value

Step 6b: BAK_ID or SK_RAND changed
+request SK from UICC
+send BAK_ID and SK_RAND

BAK_ID,SK_RAND
MBMS Identifier(s)

Step 7: UICC generates SK
+Retrieve BAK
+compue SK from SK_RAND and BAK
+send SK to ME

SK

Step 7 (cont)
+Decrypt content using new SK value

Encryption

- Step 3. SK Generation
  - The SKM creates SK from the current BAK and a random value SK_RAND. The SKM passes SK, SK_RAND, BAK_ID and BAK_Expire to the CE.
  - This step occurs frequently. Frequency may vary. The SKM should be instructed how often a new SK should be generated.

QUALCOMM

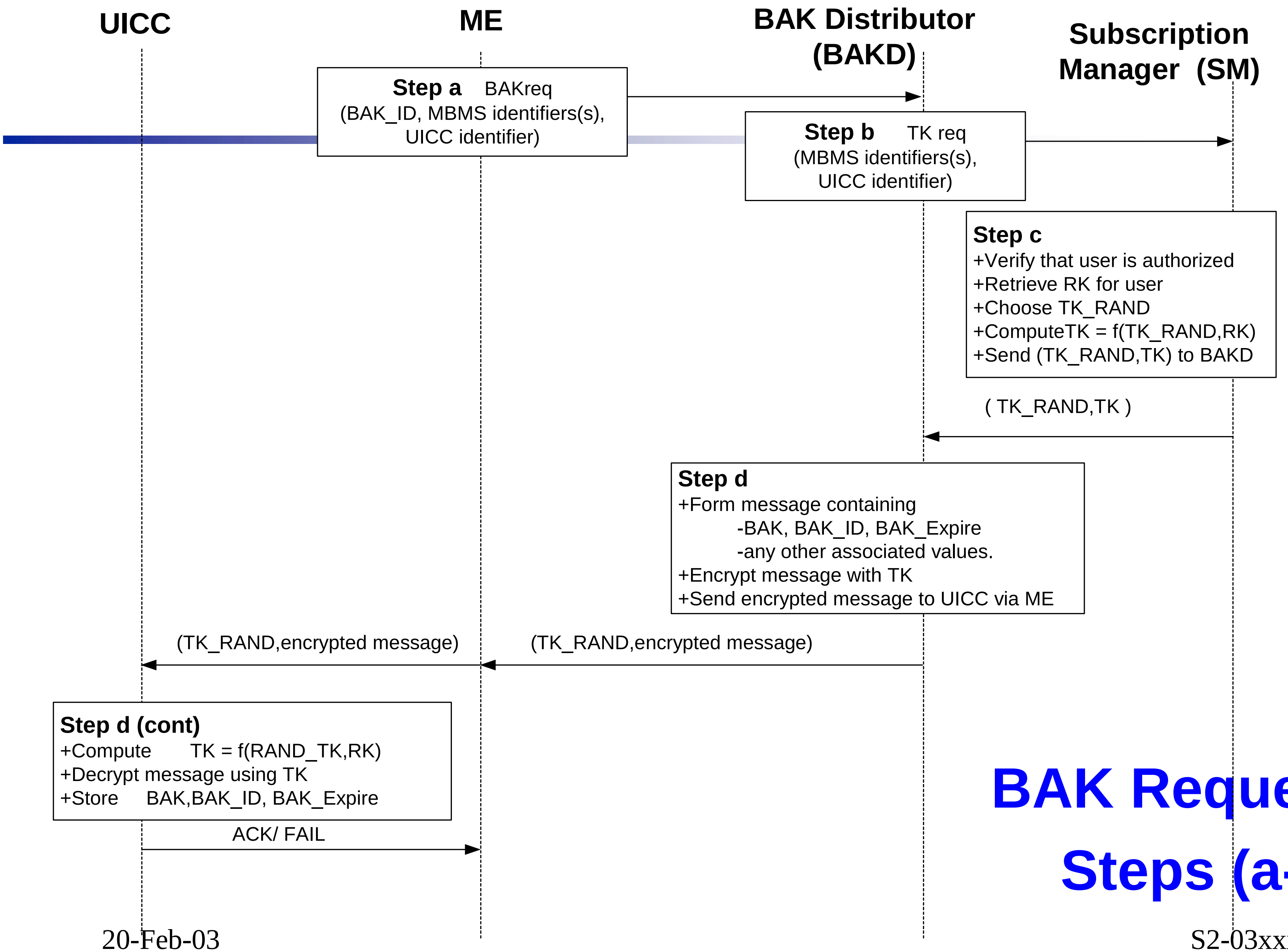
Encryption of Content

- Step 4. CE receives content
  - This step may be a continuous process
- Step 5. Encryption
  - CE encrypts content using SK and sends the encrypted content to the UE via the serving system(s). The CE also includes SK_RAND and BAK_ID with the encrypted content.
  - This step may be a continuous process. The CE uses a new SK when instructed by SKM

Decryption of Content

- **Step 6.**
  - **a. If BAK_ID and SK_RAND are unchanged from the last received content, the ME decrypts the content using SK currently assigned to that content stream and passes the result to the user application;**
  - **b. If BAK_ID or SK_RAND have changed, the ME requests a new SK from the UICC, including the BAK_ID and SK_RAND.**
  - **This step may be a continuous process.**
- **Step 7. UICC Generates SK**
  - **UICC generates SK from BAK and SK_RAND, and returns SK to the ME, which decrypts the content and (pending on step 6a) passes the result to the user application.**
  - **Usually occurs only when SK changes**

BAK Request Steps (a-d)



UICC
ME
BAK Distributor (BAKD)
Subscription Manager (SM)
(TK_RAND,TK)
(TK_RAND,encrypted message)
(TK_RAND,encrypted message)
ACK/ FAIL

Step a BAKreq
(BAK_ID, MBMS identifiers(s), UICC identifier)

Step b TK req
(MBMS identifiers(s), UICC identifier)

Step c
+Verify that user is authorized
+Retrieve RK for user
+Choose TK_RAND
+ComputeTK = f(TK_RAND,RK)
+Send (TK_RAND,TK) to BAKD

Step d
+Form message containing
-BAK, BAK_ID, BAK_Expire
-any other associated values.
+Encrypt message with TK
+Send encrypted message to UICC via ME

Step d (cont)
+Compute TK = f(RAND_TK,RK)
+Decrypt message using TK
+Store BAK,BAK_ID, BAK_Expire

BAK Request

- Occurs when the UE determines that a new BAK is required
- Step a. The ME sends BAK request to BAKD
  - Includes the BAK_ID of the BAK requested.
  - May include authentication information based on RK
- Step b. BAKD requests a TK from the SM
- Step c. SM Generates TK
  - If user is authorized to access this MBMS service, then the SM generates TK from a random value TK_RAND and RK.
  - TK_RAND may be generated by the BAKD, or by the SM. TK_RAND may also be used as a challenge in the authentication process described in step a.
  - The SM sends TK and TK_RAND to the BAKD.

BAK Request: cont.

- **Step d. The BAKD encrypts BAK with TK, and sends to UICC via ME**
  - **Includes TK_RAND, and other associated values BAK_ID and BAK_Expire.**
  - **UICC first forms TK from TK_RAND and RK…**
  - **.. and then decrypts the encrypted BAK with TK to form BAK etc.**
  - **The value of BAK and its associated values are stored in the UICC**

Comments on BAK

- BAKG controls how often BAK value changes
  - If BAKG is in BM-SC, then the network operator controls how often BAK value changes
  - Regular changes are preferable since BAK_Expire can be more accurate
  - If desired, BAK can be changed in ad hoc manner: e.g., when users leave a multicast group

3GPP TSG SA WG3 Security — S3#26 **S3-030040**
25 - 28 February 2003
Sophia Antipolis, France

CR-Form-v7

CHANGE REQUEST

⌘ **33.246** **CR** **CRNum** ⌘ **rev** **-** ⌘ Current version: **0.0.3** ⌘

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⌘ ☒ ME ☒ Radio Access Network □ Core Network ☒

Title:	⌘	Explanation of BAK-based key management		
Source:	⌘	QUALCOMM		
Work item code:	⌘	MBMS	***Date:*** ⌘	25/02/2003
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:

2	*(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)*

Reason for change:	⌘	Description of key management for MBMS.
Summary of change:	⌘	Add appropriate defintions and abbreviations in Section 3. Update MBMS architecture Figure 1 inSection 4, and add MBMS functional architecture in Section 4.2. Add description of security mechanisms in Section 6.
Consequences if not approved:	⌘	

Clauses affected:	⌘	3, 4, 6				
		Y	**N**			
Other specs affected:	⌘		X	Other core specifications	⌘	
			X	Test specifications		
			X	O&M Specifications		
Other comments:	⌘					

3 Definitions, symbols and abbreviations

Delete from the above heading those words which are not applicable.

Subclause numbering depends on applicability and should be renumbered accordingly.

3.1 Definitions

For the purposes of the present document, the terms and definitions given in 3GPP TR 21.905 [1]], 3GPP TS 22.146 [2] and the following apply.

Broadcast Access Key: A 128-bit key that provides access to the content of a particular multicast service/session for a certain amount of time (for example, one day, week or month). Each encrypted multicast service should have a different BAK value. Each BAK should have the associated BAK_ID, BAK_Expire:

Broadcast Access Key Identifier: A sequence number that identifies which value of BAK is currently being used to generate SK values for a particular multicast service. For a particular BAK, the corresponding value of BAK_ID is the same for all users.

Broadcast Access Key Expire: Indicates when the BAK will expire. This may be indicated by the time when the BAKwill expire, or by the Delta time left until the BAK expires, in which case it is calculated when a new BAK is received. Therefore, there may be multiple BAK_Expire values associated with the same BAK. The UE is expected to request a new BAK value before the BAK_Expire time expires.

Broadcast Access Key Distributor: (functional entity): Obtains authorization and TK from SM and encrypts BAK for delivery to UICCs.

Broadcast Access Key Generator: (functional entity): Generates BAK, determines BAK_ID and BAK_Expire, and delivers them to the BAKD and SKM. BAKs should be generated in a secure manner and appear random.

Content: Multicast data.

Content Encryption: (functional entity): Encrypts content, and sends the encrypted content to the UE via the cellular system.

Content Provider: A third party content provider.

Registration Key: provisioned in the UICC and SM prior to joining a multicast service. Each SM should have a unique 128-bit RK for each UICC.

Short-term Key: The CE to encrypt the multicast data using the 128-bit short-term key (SK). The UE decrypts the multicast data using SK. The SK is calculated in the SKM and the UICC. The SK is derived from SK_RAND and BAK.

Short-term Key Random Number: used to calculate a unique SK

Short-term Key Manager: (functional entity): Generates SK using BAK and SK_RAND and passes SK, SK_RAND and BAK_ID to the CE.

Subscription Manager: (functional entity): performs accounting, authentication and authorization for MBMS. The SM also calculates the TK, based on the RK, used to hide the BAK. The SM may be the subscriber's home AAA (H-AAA) or be an independent entity.

Temporary Key: A 128-bit key used by the BAKD to encrypt BAK when provisioning BAK in the UICC. The TK is obtained from the SM.

~~**example:** text used to clarify abstract rules by applying them literally (place saver to retain format).~~

3.2 Symbols

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

<symbol> <Explanation>

3.3 Abbreviations

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

BAK	Broadcast Access Key
BAK_ID	BAK Identifier
BAK_Expire:	Indicates when the BAK will expire.
BAKD	BAK Distributor (functional entity)
BAKG	BAK Generator (functional entity)
CE	Content encryption (functional entity)
CP	Content provider
MBMS	Multimedia Broadcast/Multicast Service
RK	Registration Key
SK	Short-term Key
SK_RAND	SK Random Number use to calculate a unique SK
SKM	SK Manager (functional entity)
SM	Subscription Manager (functional entity)
TK	Temporary Key

4 MBMS security architecture

MBMS introduces the concept of a point-to-multipoint service into a 3G network. A requirement of a multicast service is to be able to securely transmit data to a given set of users. In order to achieve this, there needs to be a method of authentication, key distribution and data protection for a multicast service. The point-to-point services in a 3G network use the AKA protocol (see TS 33.102 [4]) to both authenticate a user and agree on keys to be used between that user and the network. These keys are subsequently used to provide integrity protection of signalling traffic and optional confidentiality protection of both signalling and user data between the RNC and the UE.

MBMS could possibly use the AKA procedure to authenticate the user. It requires its own key management/distribution process, as the same key(s) needs to be sent to a group of users. The key distribution method could rely on the point-to-point confidentiality to protect the transfer of MBMS keys. The protection of the data may also require a special mechanism.

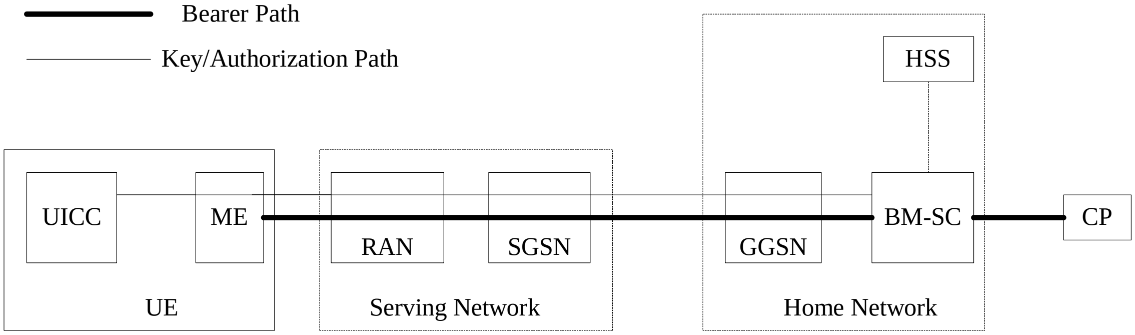


Figure 1: MBMS security architecture

Figure 1 gives an overview of the network elements involved in MBMS from a security perspective. The Broadcast Multicast – Service Centre (BM-SC) is a source for MBMS data. It could also be responsible for scheduling data and receiving data from third parties (Content Providers CP) for transmission. [Editors note: ~~(~~this is beyond the scope of the standardisation work.] The BM-SC will apply the multicast access control encryption to the multicast data. The BM-SC is also responsible for distributing keys to users that are accessing content controlled (encrypted) by the BM-SC. Signalling will be sent at the IP level.~~) for transmission~~.

Comments on Billing: Figure 1 does not include billing. The security framework described herein allows many different billing arrangements. The core assumption is that the home network (HSS) will be responsible for billing the user for access to MBMS multicast services: it may perform this billing on behalf of the serving network and/or third party content providers.

4.2 Summary of Functional Entities

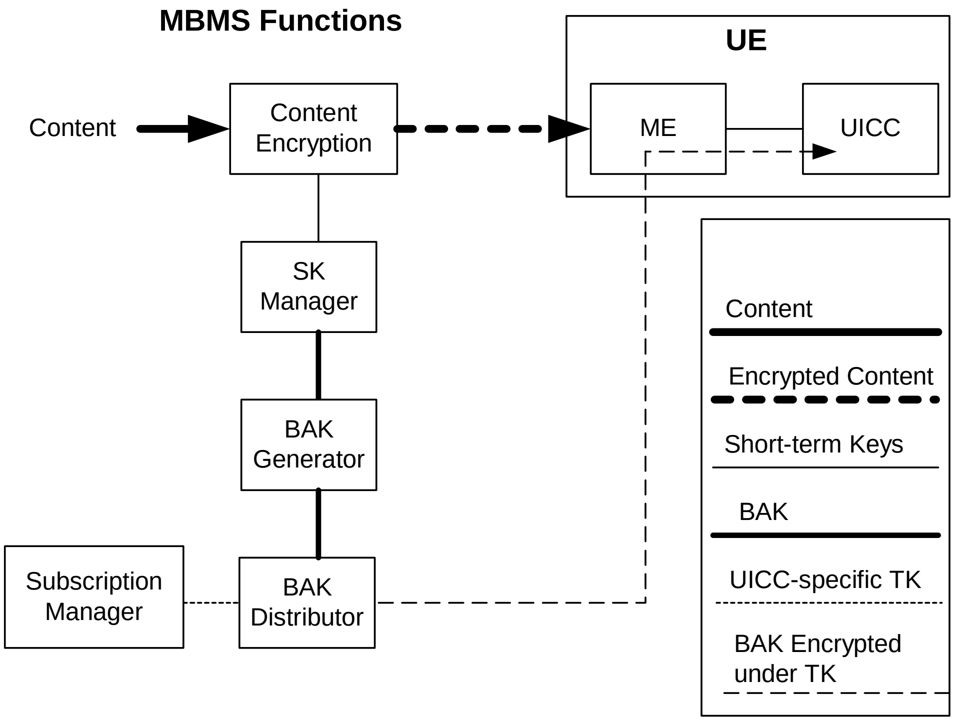


Figure 2. The MBMS Functional Architecture

Figure 2 shows the functional entities that are involved in providing multicast services. Those entities represent functions essential to support secure multicast services. They could be incorporated in one or more physical entities in the network. It should not be assumed that these functional entities correspond to separate architectural entities, so not all interfaces will require standardization. The allocation of the security functions to network entities is outside the scope of this document.

Editors note: At this point in time, it seems the SM will be associated with the HSS while the BAKG, BAKD, SKM and CE will be located in the BM-SC.

The MBMS functional entities are described in more detail below.

4.2.1 Content Path

The multicast data may be generated in the BM-SC, generated elsewhere in the home network, or generated in a third party.

Content Encryption (CE): Encrypts the multicast data using the SK values provided by the appropriate SKMs.

4.2.2 Key/Authorization Path

Subscription Manager (SM): Provide the functions of Authentication, Authorization and Accounting with respect to multicast services. The user subscribes for multicast services through one or more SM. The SM shares a Registration Key (RK) with the UICC: this key RK may be the key K used for AKA, or some other key provisioned specifically for multicast services. In order to provision a UICC with a BAK, BAKD obtains authorization (and possibly authentication) from the appropriate SM. If the user is authorized, the SM calculates a user-specific Temporary Key (TK), based on the user-specific RK, and provides TK to the BAKD.

Editor’s note: The subscription process, and the provisioning of RK in the UICC and SM are beyond the scope of this document.

BAK Generator (BAKG): Generates the Broadcast Access Key (BAK) as required, and forwards BAK to BAKD and SKM.

BAK Distributor (BAKD): Controls the distribution of the Broadcast Access Key (BAK) to the UICC of authorized users. The BAKD obtains BAK from the BAKG, and obtains authorization (in the form of TK values) from the SM.

SK Manager (SKM): Controls the updating and distribution of the Short-term Key (SK) to CE's.

User Equipment (UE): For the purposes of this document, the UE is considered as two separate entities, the UICC and ME.

> **UICC**: The UICC shares a key RK with the SM. The UICC performs all key management related to MBMS.
>
> **ME**: Mobile Equipment. Includes equipment for receiving the multicast transmission. The ME performs decryption of encrypted multicast data using SK values obtained from the UICC.

6 Security mechanisms

To counter the various threats detailed in Annex B, the multicast data is delivered encrypted to the ME, and the decryption keys (and integrity keys as appropriate) are provided only to those users who are authorized (subscribed) to receive that multicast service at that point in time. The primary focus of this document is a key management scheme for multicast services that discourages users from obtaining illegitimate access to multicast services.

The content is encrypted using a unique and frequently changing *Short-term Key* (SK). The ME decrypts the content using the same SK. The SK should be frequently changed to minimizing the impact of an authorized user sharing its SK with unauthorized users. The SK is never transmitted over the air; it is derived by the UICC from a *Broadcast Access Key* (BAK) and a random value SK_RAND that is broadcasted along with the encrypted content as shown in Figure 3.

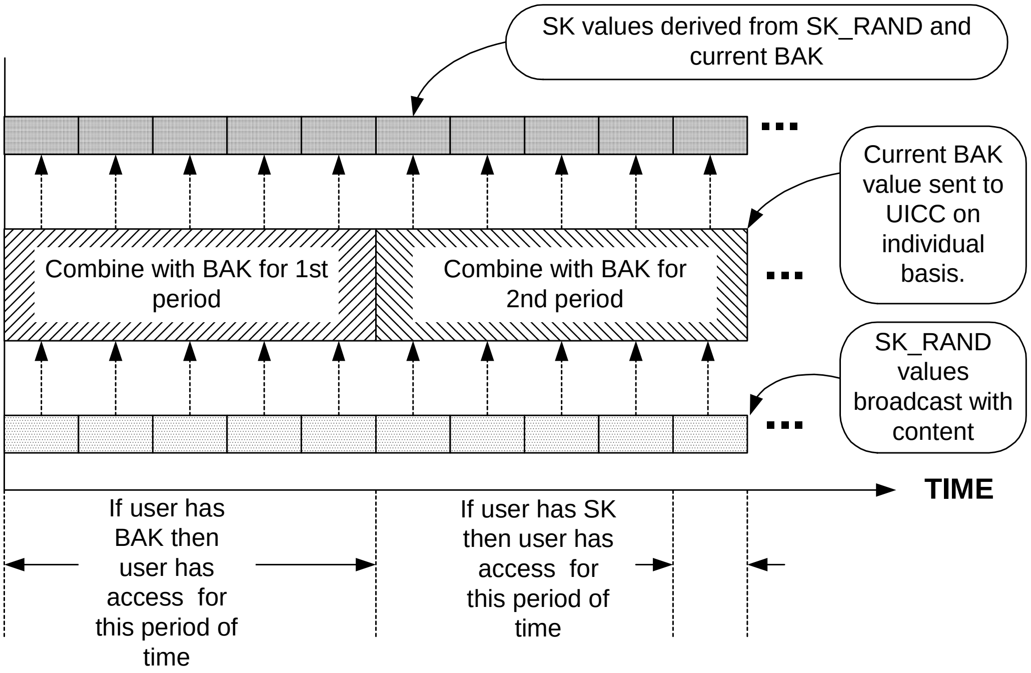

SK values derived from SK_RAND and current BAK
...
Current BAK value sent to UICC on individual basis.
Combine with BAK for 1st period
Combine with BAK for 2nd period
...
SK_RAND values broadcast with content
...
TIME
If user has BAK then user has access for this period of time
If user has SK then user has access for this period of time

Figure 3. MBMS Key Management: BAK, SK_RAND and SK

In order for the user to decrypt the necessary BAK values, the user's UICC must share an RK with an SM. [Editors note: Provisioning of RK is outside the scope of this document.] A temporary encryption key (TK) is derived from the RK by the SM, sent to the BAK Distributor (BAKD), and subsequently used by the BAKD to encrypt/decrypt the BAK.

The BAK is encrypted by the BAKD, using TK associated with that user. For a given multicast service/session, the same BAK is provided, on request, to any users subscribed to that multicast service/session, and the BAK is active for a pre-defined period of time. Once the UICC has obtained the encrypted BAK (from the BAKD), it recovers the BAK (with its own calculated TK), and then computes the SK value needed to decrypt the content. The SK is then delivered to the ME for content decryption. The BAK is never divulged to the ME, and only the UICC with a correctly pre-provisioned RK can recover the BAK.

Figure 4 shows the basic communications involved in the key distribution and multicast encryption. Many details are omitted in this diagram for the sake of clarity. The communications fall into three categories:

- Steps (1-2) occur before the UE can receive and decrypt encrypted content. Details are in Section 6.2.1.
- Steps (3-7) occur under normal circumstances, when BAK is unchanged from the previous decryption operation in the UE. Details are in Section 6.3.
- Steps (a-d) are performed to request a new BAK. Details are in Section 6.2.2.

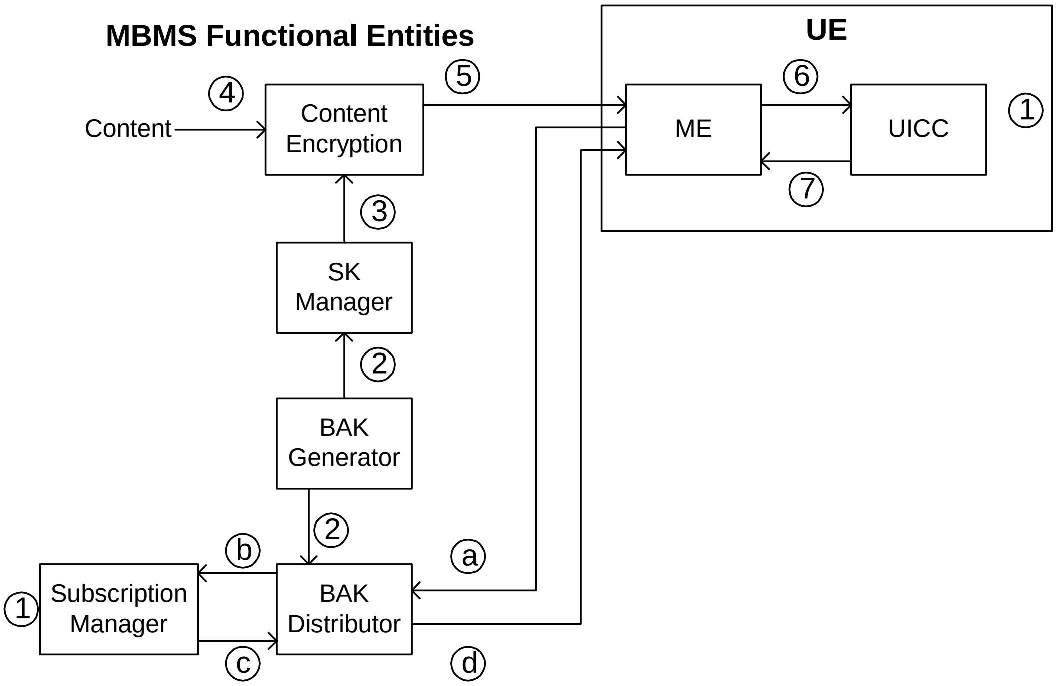


Figure 4 Logical Architecture showing functional entities involved in MBMS security.

6.1 Authentication and authorisation of a user

Editor’s note: this section will contain the details of how a user joins a particular Multicast Service

6.2 Key management

Editor’s note: this section will contain the details of how the keys are updated in a Multicast Service

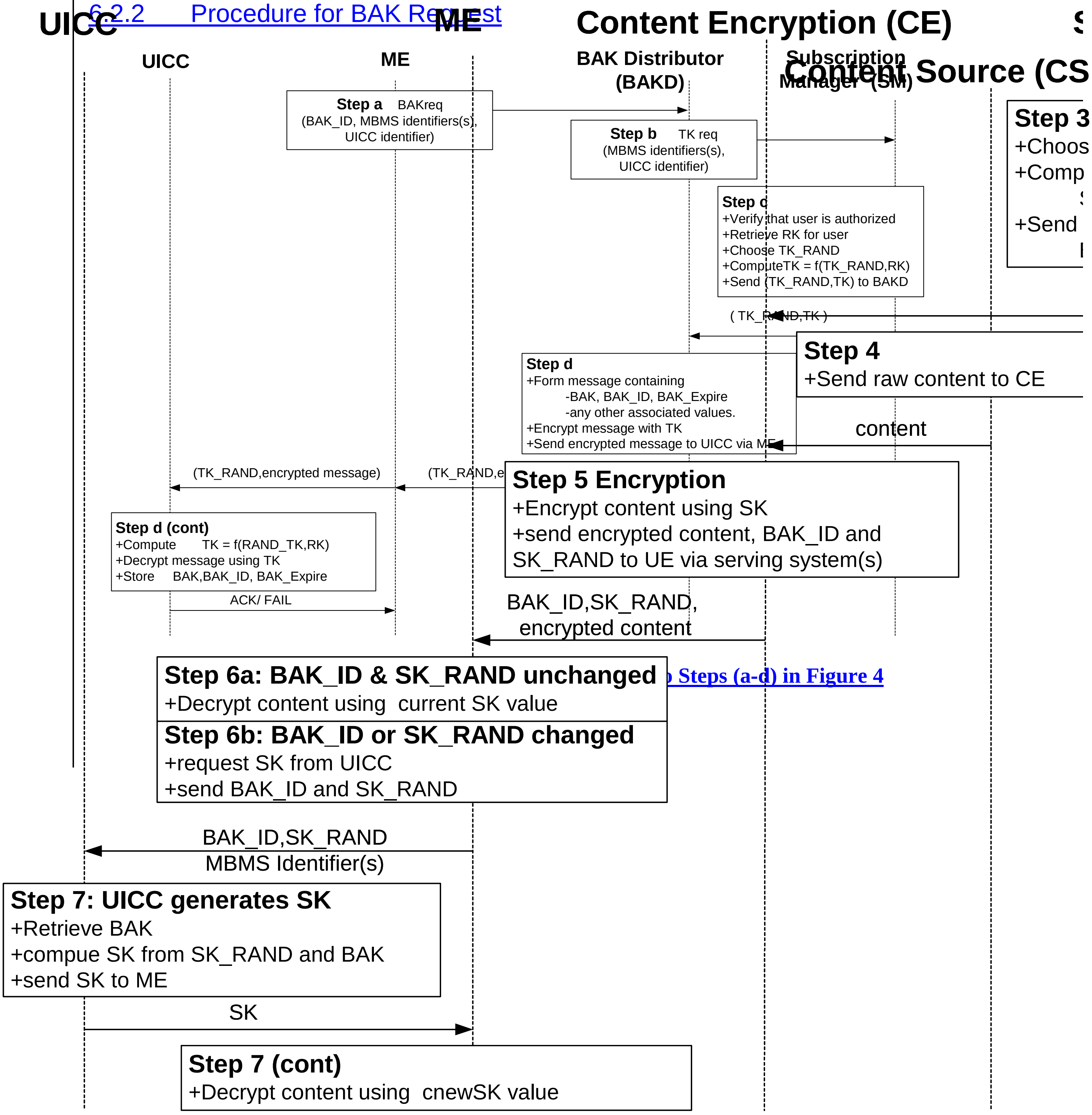
6.2.1 BAK Generation

Steps (1-2) in Figure 4 occur before the UE can receive and decrypt encrypted content:

1. The UICC and SM are provisioned with a Registration Key RK that will be the basis of authentication and key exchange with respect to MBMS.

2. The BAKG generates a value for BAK, and associates the value with an identifier BAK_ID and an expiry time (BAK_Expire). The value of BAK, along with the corresponding values of BAK_ID and BAK_Expire are passed to the SKM and BAKD.

6.2.2 Procedure for BAK Request


UICC
ME
Content Encryption (CE)
S
UICC
ME
BAK Distributor (BAKD)
Subscription Manager (SM)
Content Source (CS
Step a BAKreq (BAK_ID, MBMS identifiers(s), UICC identifier)
Step b TK req (MBMS identifiers(s), UICC identifier)
Step 3
+Choos
+Comp
+Send
Step c
+Verify that user is authorized
+Retrieve RK for user
+Choose TK_RAND
+ComputeTK = f(TK_RAND,RK)
+Send (TK_RAND,TK) to BAKD
(TK_RAND,TK)
Step 4
+Send raw content to CE
Step d
+Form message containing
-BAK, BAK_ID, BAK_Expire
-any other associated values.
+Encrypt message with TK
+Send encrypted message to UICC via ME
content
(TK_RAND,encrypted message)
(TK_RAND,e
Step 5 Encryption
+Encrypt content using SK
+send encrypted content, BAK_ID and SK_RAND to UE via serving system(s)
Step d (cont)
+Compute TK = f(RAND_TK,RK)
+Decrypt message using TK
+Store BAK,BAK_ID, BAK_Expire
ACK/ FAIL
BAK_ID,SK_RAND,
encrypted content
Step 6a: BAK_ID & SK_RAND unchanged
+Decrypt content using current SK value
Step 6b: BAK_ID or SK_RAND changed
+request SK from UICC
+send BAK_ID and SK_RAND
BAK_ID,SK_RAND
MBMS Identifier(s)
Step 7: UICC generates SK
+Retrieve BAK
+compue SK from SK_RAND and BAK
+send SK to ME
SK
Step 7 (cont)
+Decrypt content using cnewSK value

Steps (a-d) in Figure 4

Steps (a-d) in Figure 4 are performed to request a new BAK. Section 6.3 discusses possible methods by which the UE determines that a new BAK is needed.

a. The ME sends a BAK request to the BAKD, including the BAK_ID of the BAK requested. The BAK request may include authentication information based on RK, which can be used by the SM to determine that the request came from a legitimate subscriber. Section 6.2.3.3 discusses the need for authentication of BAK requests.

b. In order to send BAK to the UICC, the BAK must be encrypted to protect it against reception by other than the intended recipient. The BAKD requests a temporary key (TK) generated by the SM.

c. If the user is authorized to access this MBMS service, then the SM generates TK from a random value TK_RAND and RK. TK_RAND may be generated by the BAKD, or by the SM. TK_RAND may also be used as a challenge in the authentication process described in step a. The SM sends TK and TK_RAND to the BAKD.

d. The BAKD encrypts the value of BAK with TK, and sends the encrypted BAK, along with TK_RAND and BAK_Expire to the UICC via the ME. The UICC first forms TK from TK_RAND and RK, and then decrypts the encrypted BAK with TK to form BAK. The value of BAK and its associated BAK_ Expire are stored in the UICC. The UICC should be able to store at least two values of BAK so that a new BAK can be obtained and stored in anticipation of the expiration of the BAK lifetime.

6.2.3 BAK Management

A user can repeat the BAK request procedure with multiple multicast services. Thereafter (from a security perspective) the UE has the potential to be managing the decryption keys for multiple multicast services simultaneously.

Editor's note: The issue of associating BAK values with a particular MBMS session is FFS.

This document does not specify how an UE determines that it needs to update BAK. We assume that a means will be provided for the UE to determine that its BAK is about to expire or has expired, triggering action to perform a BAK request. Several methods are possible for accomplishing this. The specific method used is not important for the present subject.

BAK_ID. There will be times when the UICC needs to store two BAK values associated with a particular MBMS session. The reason is that the BAK may be updated in the UICC prior to the BAK actually being used to encrypt/generate SK values (see Section 6.2.3.1). Consequently, both the old BAK and the new BAK will reside in the UICC simultaneously: both being associated with the same MBMS session. The UICC must have some means to determine which of the two BAK values is being used to generate the requested SK. The recommended method is to allocate a BAK identifier BAK_ID to each BAK, in addition to whatever method is used to associate the BAK with the MBMS session. The BAK_ID is likely to be a sequence counter, or something similar. The CE will include the BAK_ID with the encrypted content in the multicast data. The UE can now distinguish if the old BAK is still being used or if the new BAK has begun to be used. To save bandwidth, the BAK_ID will consist of only a few bits.

Editor's note: A 4-bit BAK_ID is probably appropriate.

BAK_Expire. There is the potential for two different BAK values to have the same BAK_ID values. The small size of the BAK_ID results in all BAK_ID values having been used after the BAKG has generated only a relatively small number BAK values. Thereafter, new BAK values start being allocated previously used BAK_ID values. To solve this problem, the BAKD provides a BAK_Expire value along with the BAK. The UE can use the BAK_Expire to distinguish which BAK is the correct BAK. The UE can also use BAK_Expire to determine if keys have expired and need updated.

Frequency of BAK changes. The BAKG entity is responsible for changing BAK. The BAKG must know the duration of time for which BAK values are used to generate SK values (alternatively, the BAKG must know how often BAK changes). The following issues should be considered in deciding how often BAK is to be changed.

- Frequent BAK changes will provide more security.
- Frequent BAK changes will also provide greater flexibility in subscription control. We show this by example. Once a user has BAK, they can access the content for the lifetime of that BAK. Suppose the BAK is changed at the beginning of every month. If a user's subscription runs out halfway through the lifetime of

a BAK, the user will still be able to generate SK (and thus view the content) until the BAK expires. So by changing BAK only every month, the SM can only charge subscriptions from the beginning of the month to the end of the month. A user can't subscribe from the middle of one month to the middle of the next. However, if BAK changed every day, then the user could subscribe from the beginning of any day during the month.

Increasing the frequency of BAK changes should be evaluated against a possible increase in the number of times the mobile station has to retrieve new BAK values.

6.2.3.1 Updating BAK before use

It may prove beneficial to provision the UICC with BAK shortly before BAK begins being used to derive SK values. Otherwise, once the CE starts sending packets with SK derived from the new BAK, the user would experience a delay as the UE performs a BAK update. If many users are tuned in, then there will be a burst of traffic as all the MSs perform a BAK update.

To avoid such problems, the BAKD should allow an UE to obtain the new BAK shortly before the BAK changes. Different UE may have different schedules for performing BAK updates, to prevent too many MSs performing a BAK update at once.

For security reasons, BAK should be distributed as close as possible to the time of use.

6.2.3.2 Storage of BAK

It is essential to the security model used herein that the UICC does not reveal BAK. If a single UICC reveals BAK, then all security is compromised until the BAKG changes BAK.

The UICC should store BAK and related data about BAK, such as BAK_ID and BAK_Expire.

Editor's note: The issue of associating BAK values with a particular MBMS session is FFS.

The ME may store the BAK-related data (such as BAK_ID and BAK_Expire, but not BAK), to save requesting this information from the UICC.

6.2.3.3 Authenticating BAK Requests

An adversary cannot obtain BAK by performing a BAK request while impersonating a subscribed user. Only the subscribed user will be able to derive TK from RAND_TK, and thus extract BAK. For this reason, the BAKD does not need to authenticate BAK requests in order to protect BAK.

If, however, other control functions or accounting data are included with the BAK request, then it is necessary to prove the authenticity of the data source. It may also be desirable to authenticate requests to prevent malicious generation of excess network traffic.

Editor's note: The procedures for authenticating the request are for further study.

Note that even though authentication of BAK requests may not be needed, there is still a need for authorization of the request. Such authorization must be specific to the multicast service to which the requested BAK applies.

6.2.4 Security Algorithms for Key Management

Many of the security algorithms for key management should be standardized.

6.2.4.1 Encryption of BAK

BAK encryption (under key TK) shall be standardized.

6.2.4.2 Management of TK

6.2.4.2.1 TK_RAND

TK_RAND shall be 64 bits in length.

TK_RAND shall be generated in a manner that minimizes the probability that the same TK_RAND is used to create a TK for different BAK encryptions destined to the same terminal. The probability should be the same as for random selection of TK_RAND.

6.2.4.2.2 TK Generation

TK shall be 128 bits in length.

TK shall be generated from TK_RAND and RK using a secure one-way function chosen to minimize the likelihood that an adversary can obtain TK with knowledge of TK_RAND alone, and to minimize the likelihood that an adversary can obtain RK with knowledge of TK_RAND and TK.

6.2.4.3 Management of SK

6.2.4.3.1 SK_RAND

SK_RAND shall be 32 bits in length.

SK_RAND shall be generated in a manner that minimizes the probability that the same SK_RAND is used to create an SK for different content encryptions destined to the same terminal. The probability should be the same as for random selection of SK_RAND.

SK_RAND shall be generated in a manner that minimizes the probability that an adversary can guess the next value of SK_RAND with knowledge of previous values. This requirement makes it more difficult for a modified ME shell to compute future SK values and distribute the values to other MEs.

6.2.4.3.2 SK Generation

SK shall be 128 bits in length.

SK shall be generated from SK_RAND and BAK using a secure one-way function chosen to minimize the likelihood that an adversary can obtain SK with knowledge of SK_RAND alone, and to minimize the likelihood that an adversary can obtain BAK with knowledge of SK_RAND and SK.

6.3 Protection of the transmitted traffic

Editor’s note: this section will contain the details of how traffic is protected

The transmitted traffic is encrypted with the primary purpose of controlling access so that the serving network can bill users that access the multicast data.

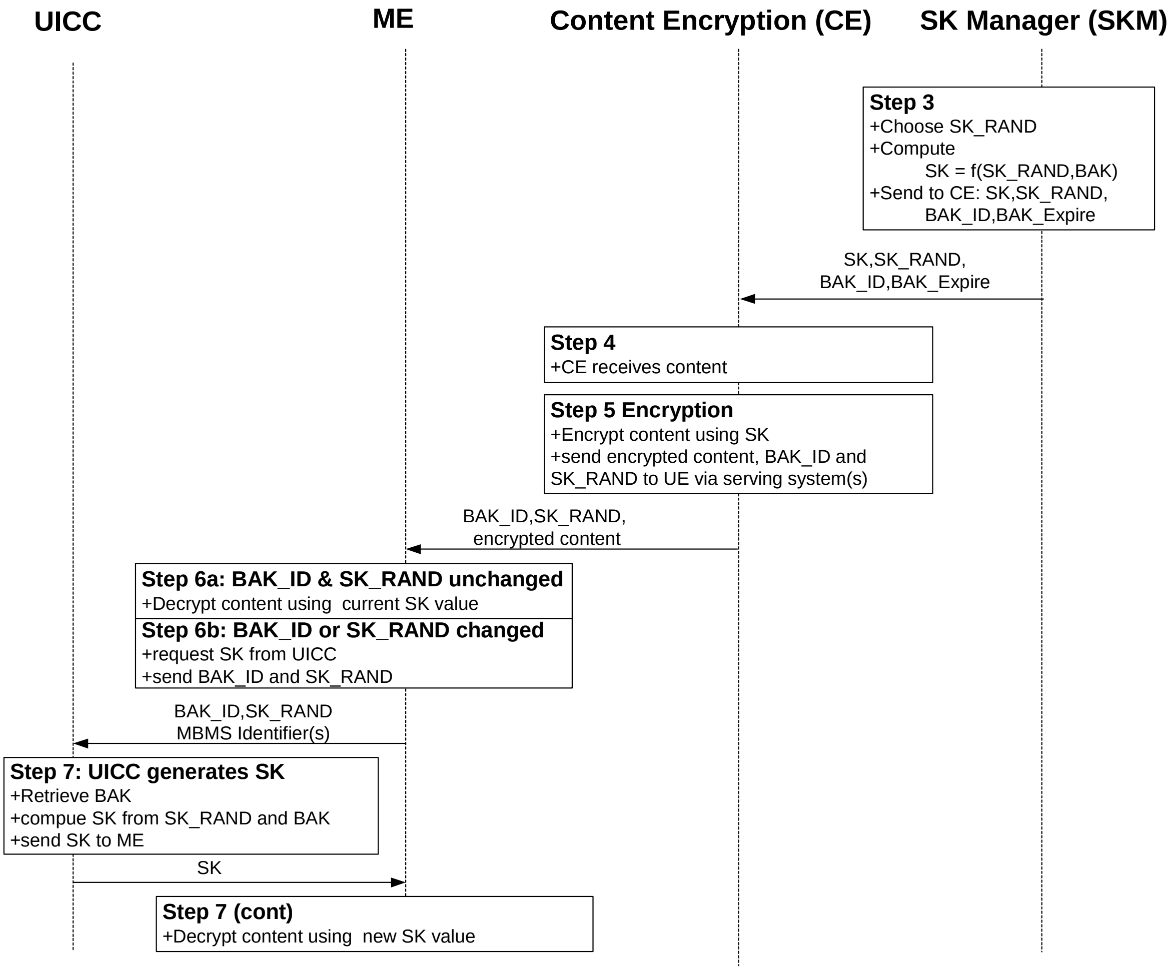


Figure 6 Encryption and Decryption. Corresponds to Steps (3-7) in Figure 4

The steps (3-7) occur under normal circumstances, when BAK is unchanged from the previous decryption operation in the UE.

3. The SKM creates SK from the current BAK and a random value SK_RAND. The SKM passes SK, SK_RAND, BAK_ID and BAK_Expire to the CE.

4. The CE receives the content.

5. The CE encrypts the content using SK and sends the encrypted content to the UE via the serving system. The CE also includes SK_RAND and BAK_ID with the encrypted content.

6. The ME receives the encrypted content, and takes action as follows:

   a. If BAK_ID and SK_RAND are unchanged from the last received content, the ME decrypts the content using the value of SK currently assigned to that content stream and passes the result to the user application;

   b. If BAK_ID or SK_RAND have changed, the ME requests a new SK from the UICC, including the BAK_ID and SK_RAND.

Editor's note: The issue of associating BAK values with a particular MBMS session is FFS.

7. The UICC generates SK from BAK and SK_RAND and returns SK to the ME, which decrypts the content and (pending on step 6a) passes the result to the user application.

6.3.1 Encryption Algorithm

The data encryption algorithm for multicast services must be standardized.

The format of the encrypted data may depend on the architecture, and is currently beyond the scope of this document.

Annex C (informative): Explanation of Key Management

To access the multicast content, a user must have the current decryption keys. The UICC is not powerful enough to decrypt the content so the ME must perform the decryption. This implies that the decryption keys must be stored in the ME, which cannot be considered a secure storage device. It must be considered that eventually an attacker may find a way to extract the current decryption key from the ME. An attacker who is a subscribed user will then be able to distribute the decryption key to other non-subscribed users. In summary, the need to store decryption keys in in-secure memory makes it impossible to design a scheme where non-subscribed users CANNOT access the data.

We must recognize that the most we can do is dissuade the potential market (those users for which the service is targeted) from using illegitimate means to access the content. The potential market is a user that desires **cheap** access to multicast services while being **mobile**. This means that we should not be so concerned about attacks that are expensive to mount (that is, expensive on a per-user basis) and we should be so concerned about attacks when the user is not mobile. For example, we should not be concerned with attacks that require the user to perform a frequent data download of a key; mobile data downloads will be more expensive than the subscriptions.

Assuming the primary threat is subscribed users distributing decryption keys to non-subscribed users, the solution is for the decryption key to change frequently and in an unpredictable manner. The challenge is achieving this while minimizing the transmission overhead required for key distribution. The solution described herein is to distribute a Broadcast Access Key (BAK) to each user individually, and for many decryption keys to be derived using the BAK and public information sent with the encrypted content. The BAK is stored in the UICC.

In a flat-rate subscription scheme, the user will have paid for access to the associated MBMS data, whether or not the user accesses that data. The BAK-based key management is perfectly suited to such a scheme.

In a usage-based subscription scheme, the user should only pay for the MBMS data accessed by the user. The BAK-based key management is adequate for usage-based schemes as well. If BAK is changed frequently, then the number of BAK values requested by their UICC forms a good measurement of the usage. In a usage-based scheme, the user can be charged for the number of BAK values requested by their UICC.

C.1 Motivation for Establishing Registration Keys

Recall that the purpose of the MBMS security is to prevent unauthorized (non-subscribed) users from accessing subscription-based content. Consequently, an unauthorized user should not be able to obtain BAK values. An unauthorized user may try to obtain BAK from a BAK Distributor (BAKD) by providing the identity of a subscribed UICC during a BAK request. If the unauthorized user can control what key is used to encrypt BAK, then the unauthorized user will be able to decrypt the packets sent back by the BAKD, and obtain BAK.

The best way to counter this attack is to encrypt the BAK using a key known only to the correct UICC (that is, the UICC identified in the BAK request). In this case, any other UICC will not be able to decrypt the packets sent back by the BAKD. This provides implicit authentication of the UICC.