3GPP TSG SA WG3 (Security) meeting #25 **S3-020501**

8-11 October 2002

Munich, Germany

Source:	**Greg Rose, QUALCOMM**
Title:	**Draft 3GPP2 Broadcast / Multicast Service security specifications**
Document for:	**Information and review**
Agenda Item:	**7.19**

3GPP2 is examining a hierarchical key management system for their Broadcast/Multicast Service (BCMCS) that is largely analogous to MBMS in 3GPP. Attached in the zip file are the current Stage 2 document S.P0083 “Broadcast Security Framework” and a corresponding powerpoint presentation. These are presented for the information of SA3.

3GPP2 S.P0083

Version 0.1

Version Date: 10 September 2002

3RD GENERATION PARTNERSHIP PROJECT 2 "3GPP2"

Broadcast Security Framework

EDITOR

Philip Hawkes
QUALCOMM Australia
+(61-2) 9817-4188
phawkes@qualcomm.com

REVISION HISTORY

REVISION HISTORY		
Rev. 0.1	*First draft*	*11 September 2002*
Rev. 0.2		
Rev. 0.3		
Rev. 0.4		

Table of Contents

1 Introduction and Scope

This document defines the security framework for the Broadcast Service Phase of Broadcast/Multicast Services (BCMCS).

2 References

3 Definitions and Abbreviations

AAA Authentication, Authorization, and Accounting.

AAA-H holds voice/data subscriptions. The AAA-H and UIM (of users with a voice/data subscription) share an A-key K. The AAA-H is part of the home Service Provider's network.

AC Authentication Center. This is an ANS-41 entity performing authentication and key management functions similar to those performed by an AAA.

BAK Broadcast Access Key: Provides access to the content of a particular BCMCS for a certain amount of time (for example, one day, week or month). Each encrypted BCMCS should have a different BAK value. Each BAK should have the following associated values:

BAK_ID: BAK identifier. A sequence number that identifies which value of BAK is currently valid. For a particular BAK, the corresponding value of BAK_ID is the same for all users.

BAK_lifetime: Time left until the will BAK expire. This time will depend on when the MS requests the BAK. Thus, when two different MS's request the same BAK, the corresponding value of BAK_lifetime will change depending on when those two MS's request that BAK.

BAK_TTU: BAK time-to-update. This is the amount of time left until the MS should request a new BAK. The value of BAK_TTU should be "randomized" so that not all MS's request BAK at the same time. Hence For a particular BAK, the corresponding value of BAK_TTU will change from MS to MS.

BCMC Content Broadcast-Multicast Content, also refered to simply as **Content**. The content is the data being broadcast. Typically, this is expected to be audio-visual data.

BCMCS and BCMC service Broadcast-Multicast Service. BCMCS is the name of the system for offering broadcast and multicast services. The term "BCMCS" or "BCMC Service" is also used to describe a single broadcast stream: equivalent to a TV channel. For example, the document may refer to a "CNN BCMCS" or "local news BCMC service".

BCMCS Control An entity in the RAN. The BCMCS Control has many functions relating to control of BCMC services and key management.

BCMCS_ID A number allocated by the BCMCS Control for local identification the BCMCS. The BCMCS_ID is only valid in the cells controlled by that BCMCS Control.

BCMCS Profile When a user has a BCMCS subscription, the BCMCS Profile is a description of the BCMC services to which the user has subscribed. The user

BCMCS Profile is stored in the MS and the subscription entity (AAA-H or BCMCS Subscription Server).

BCMCS-SS or SS BCMCS Subscription Server: a third-party subscription entity (that is, a subscription entity that is not the user's AAA-H). The SS may part of the CS. Alternatively, the CS may allow the SS to offer subscriptions to BCMC services offered by the CS (as part of a business arrangement).

Content_ID Content Identifier: A value used to identify the type of content offered on a particular BCMC service. For example, there may be a special Content_ID reserved for major broadcasters (such as CNN) or common services (such as news, stock updates, traffic information).

CS Content Server: Provides the data for the service. A CS may include data taken from other content servers, and include it in their content. A CS may include the following entities:

BCMCS Security Manager The Security Manager is required when the user's subscription is not held with the home network. The Security Manager generates and encrypts BAK for provisioning into UIMs. This entity also passes BAK on to the entity performing the content encryption.

Content Encryption Used when end-to-end encryption is applied.

Content Source Generates the un-encrypted content.

EBAK An encrypted value of BAK. A temporary key TK is used when encrypting BAK.

ESK An encrypted value of SK. A BAK is used as the key when encrypting SK.

K A shared key held in the UIM and AAA-H.

Local CS A local CS is one that provides BCMC content to the system in which the user is currently located. That is, a local CS provides content to the visited SP.

MS Mobile Station: For the purposes of this document, the MS is considered as two separate entities, the UIM and ME.

UIM: User Identity Module (UIM): The UIM is a low power processor that contains secure memory. The UIM may be removable (like a SIM card) or part of the MS itself.

ME: Mobile Equipment: The ME contains a high power processor, but no secure memory.

PDSN Packet Data Serving Node: Interfaces between the Internet and the RAN.

RAN Radio Access Network.

RK Registration Key: stored in the UIM and SS. The RK is derived from the shared key K held in the UIM and AAA-H. Each SS should have a unique RK each UIM, and similarly no two SS's should use the same RK.

RK Establishment The process by which a UIM and BCMCS Subscription Server determine a shared Registration Key (RK).

SA Security Association: A listing of the parameters (such as the key) required to process an IPSec packet. Each SA is index by destination address and Security Parameter Index (SPI).

SDP Session Data Parameters: Parameters required for processing the current content.

SK Short-term Key: also called the decryption key. The BCMC content is encrypted and decrypted using SK.

SK_ID: SK identifier. The SK_ID is used by the ME to determine when SK changes.

SK Lifetime: The short-term key sis changed frequently. The SK lifetime tells the encrypting entity how often SK should be changed.

SP Service Provider: a network providing voice/data services.

Visited SP The serving network in which the MS is currently located.

Home SP The network holding the user's voice/data subscription.

SPI Security Parameter Index: used to index a security Association (SA) in IPSec.

SUB_ID BCMCS Subscription identifier. The SUB_ID is allocated to the UIM by the subscription entity when the user first subscribes with the subscription entity. The SUB_ID includes information identifying the subscription entity.

Subscription Data A description of the BCMC services to which the user has subscribed through a particular subscription entity. The subscription data is stored in the MS and the subscription entity (AAA-H or BCMCS Subscription Server). The subscription data may relate to multiple BCMC services. For example, the subscription data may be a list of the BCMCS_ID and Content_ID values corresponding to BCMC services that the user has subscribed to, along with a description of the dates or times for which the subscription is valid.

Subscription Entity An entity that holds BCMCS subscriptions.

TK Temporary Key: used by the BCMCS Security Manager or BCMCS Control to encrypt BAK when provisioning BAK in the UIM. The TK is obtained from the subscription entity.

Voice/Data Subscription: This is a subscription to normal voice and data services. Voice/data subscriptions are held in the AAA-H (in the home service provider) of the user. Users whose voice/data subscriptions are in a particular AAA-H are said to be voice/data subscribers of that AAA-H.

4 Overview of Broadcast Service

4.1 Introduction to BCMCS

The aim of BCMCS is to provide the following service. Entities, called Content Servers (CSs), will provide content to participating mobile Service Providers (SPs). The content is envisioned to be audiovisual data. A CS may be part of the serving network, but this is not necessarily so. The SP will transmit this content on a particular physical channel. If the content is offered for free, then any user can access this channel to view/process the content. If access to the channel is subscription based, then while any users can tune into the physical channel, the content will be encrypted so that only the subscribed users will be able to view/process the content. This document addresses the security needs for subscribed broadcast services.

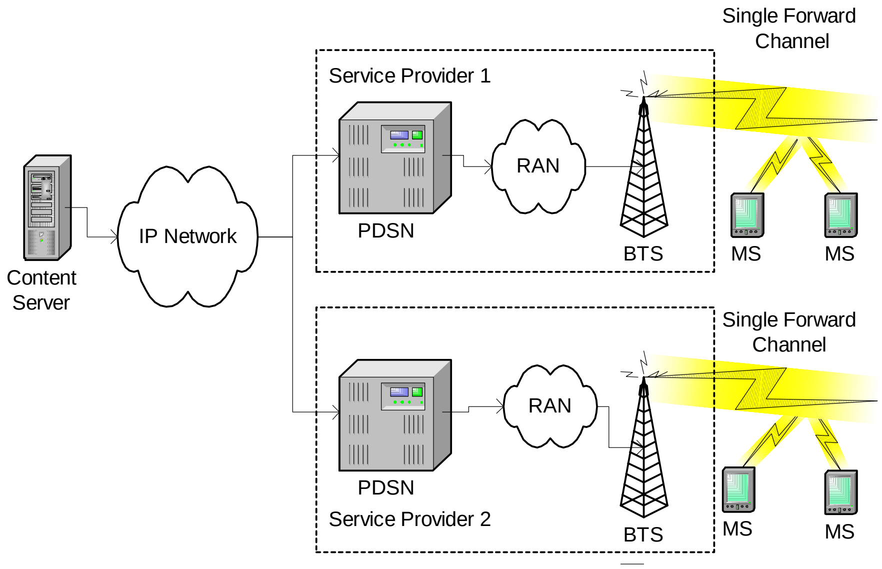


Figure 4.1 Broadcast/Multicast Service

A single CS may provide more than one broadcast-multicast service to a particular carrier, and may provide the services to more than one carrier. To complicate matters, the users may subscribe through some one other than the CS: for example, the user may subscribe through their carrier. An entity with which the user may subscribe is called a *subscription entity*. A third party subscription entity is called a *Subscription Server* (SS). The user may subscribe to more than one subscription entity. These subscription entities will provide service authorization and are an integral part of the key management.

We must recognize that the most we can do is dissuade the potential BCMCS market (those users for which the service is targeted) from using illegitimate means to access the content. This document presumes that the BCMCS market is for

- Users that are mobile.
- Users that want quality service that is easy to access.

4.2 Summary of Key Management

For the purposes of this document, the Mobile Station (MS) is considered as two separate entities, the User Identity Module (UIM) and the Mobile Equipment (ME). The UIM is a low power processor that contains secure memory. The UIM may be removable (like a SIM card) or part of the MS itself. The ME contains a high power processor, but no secure memory.

This document assumes that the only insecure links are over the air (between the BSC/PCF and Mobile Station), and between the UIM and ME. That is, the communication is secured between entities in the RAN, entities in the CS and the SS.

The main threat addressed in this document is the threat of a user obtaining access to the content without paying a subscription fee (where required). To counter this threat, the content is encrypted and decryption keys provided only to those users who have subscribed. The primary focus of this document is addressing the key management for such a scheme. If the content is not encrypted, then the MS need only listen to the appropriate logical broadcast channel: no keys are required.

The content is encrypted at either the link layer or IP layer using a frequently changing *Short-term Key* (SK). The ME decrypts the content using SK. SK is changed frequently to prevent a “rogue shell” from sending SK to other terminals for use in receiving the broadcast, thereby providing many with service with only a single paying subscription. The value for SK is not transmitted. If link-layer encryption is applied, then an encrypted value of SK is broadcast, whereby a *Broadcast Access Key* (BAK) is required to decrypt and obtain SK from the broadcast. If IP-layer encryption is applied, then SK is derived from BAK and a 28-bit random value. The random value is broadcast along with the content encrypted using SK, so the MS can derive SK from BAK the broadcast random value. The BAK resides in the UIM, requiring the presence of the UIM in order to receive broadcast service. The current BAK is the same for all subscribers to that channel, and the BAK provides access for a period of time determined by the operator. Thus, once the UIM has obtained the BAK, the UIM can compute the SK values needed for the ME to decrypt the broadcast.

Multiple SK values are derived from the same BAK value. An example is shown in Figure 4.2. In this example, only three decryption keys are derived from each BAK. In practice, there may be hundreds or thousands of decryption keys derived from a single BAK.

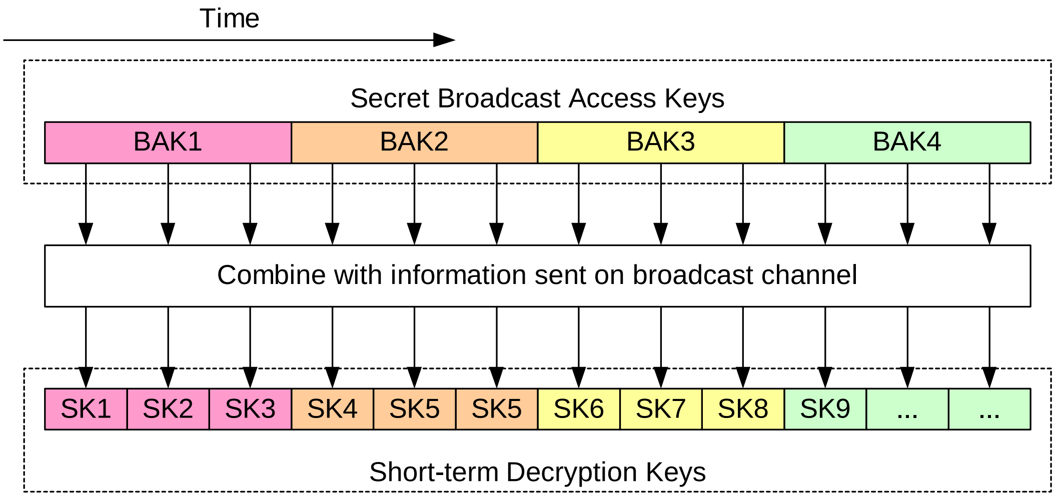

Time
Secret Broadcast Access Keys
BAK1
BAK2
BAK3
BAK4
Combine with information sent on broadcast channel
SK1
SK2
SK3
SK4
SK5
SK5
SK6
SK7
SK8
SK9
...
...
Short-term Decryption Keys

Figure 4.2. An example of deriving many decryption keys from one BAK by combining with information sent on the broadcast channel

When the user first subscribes to a particular Subscription Server (SS) for the first time, the AAA-H provisions a Registration Key (RK) to the UIM and SS. This key is the basis for future transport of keys involved with that Subscription Server.

Either a BCMCS Security Manager in the CS or a BCMCS Control in the visited SP provisions the BAK in the UIM. The BAK is encrypted using Temporary Keys (TK) derived from RK, so only the correct UIM can decrypt to obtain the BAK.

4.3 Preliminaries

4.3.1 BCMCS_ID and Content_ID

BCMCS_ID Each BCMCS is identified locally by a BCMCS_ID value. The BCMCS_ID is only valid in the local area and is allocated by the visited BCMCS Control. If a subscription only applies to the local network, then the subscription entity may specify specific BCMCS_ID values for services to which the user is subscribed.

Content_ID A Content_ID is used to identify the content being sent by a CS. For example, around the world there may be many CS sending "FOX Sports". There would be a global Content_ID value corresponding to "FOX Sports". There may be Content_ID values for local news and other types of broadcasts. The idea is that the same Content_ID value in two separate locations will identify the same "type" of content.

When a BCMCS corresponds to a service with a global Content_ID, this Content_ID will be broadcast on overhead channel, along with the BCMCS_ID and other parameters. The purpose of a Content_ID is twofold. Suppose the user has a global subscription to content with certain Content_ID values. The MS can ascertain those BCMC services that the user is subscribed to, using the Content_ID values in overhead message. This is the first purpose of the Content_ID. When the visited BCMCS Control or BCMCS Security Manager obtains the subscription data, the subscription data will include the Content_ID values for which the user is subscribed. The BCMCS Control or BCMCS Security Manager can use these values for deciding if the user is subscribed to the requested service. This is the second purpose of the Content_ID.

4.3.2 Business Relationships

The following pairs of entities are assumed to have some sort of business agreement and therefore "trust" each other.

- **The visited Service Provider (SP) and home SP**. Standard roaming agreement.

- **The home SP and third-party Subscription Server (SS)**. This agreement is required since the Subscription Servers trust the home Service Provider to establish Registration Keys (RK). The home Service Provider may charge the Subscription Servers for establishing RK.

- **The visited Service Provider and local Content Servers.** The visited Service Provider will charge the local Content Servers for the transmission of the broadcast. If the visited Service Provider is provisioning BAK (through a BCMCS Control), then the visited SP may also charge the local Content Servers for this service.

- **The local Content Server and subscription entity.** If a local Content Server provides a user with access based on the user's subscription to a subscription entity, then it is assumed that the Content server has an agreement with the subscription entity. The Content Server will expect payment from the subscription entity. The subscription entity may be the home Service Provider or a third-party Subscription Server.

- **(In some cases) The visited Service Provider and third-party Subscription Server.** If the visited Service Provider is performing BAK management, then the Subscription Server and visited Service Provider may have an agreement that the visited Service Provider will manage BAK values for users subscribed through the Subscription Server.

4.3.3 Link-Layer Encryption

(YET TO BE FINISHED)

4.3.4 IPSec

One option for end-to-end encryption is to use IPSec Encapsulating Security Payload (ESP) in transport mode. The security parameters, such as the encryption key and the encryption algorithm, are stored as a *security association*, which is indexed by the destination address and a 32-bit value called the *Security Parameter Index* (SPI).

IPSec has not previously been used for this type of broadcast service. Consequently, the key management here is different from that usually applied to IPSec. For BCMCS, the SK is derived from the BAK and SPI in the header of that IPSec packet. This key management does not deviate from IPSec per se, but it does use the SPI in a non-standard way. We feel that this style of key management is necessary in offering efficient broadcast services.

5 Requirements

5.1 Summary

At this point in time, it is difficult to predict what business models will be best suited to BCMCS. This document provides options for key management and encryption. These options should accommodate a large range of business models.

Section 5 is organized as follows. Section 5.2 describes the BCMCS Security framework in a very general manner, defining entities and showing where entities lie within thte network. Section 5.3 describes the RK Establishment process. Section 5.4 discusses the subscription process. Section 5.5 describes BAK Management by a BCMCS Control and BAK management by a BCMCS Security Manager. Section 5.6 describes the processes of SK generation and content encryption when encryption is at the link-layer and when encryption is performed at the IP layer. Section 5.7 describes the functionality of the entities in the BCMCS security framework. Section 5.8 contains some useful tables.

5.2 High-level Architecture

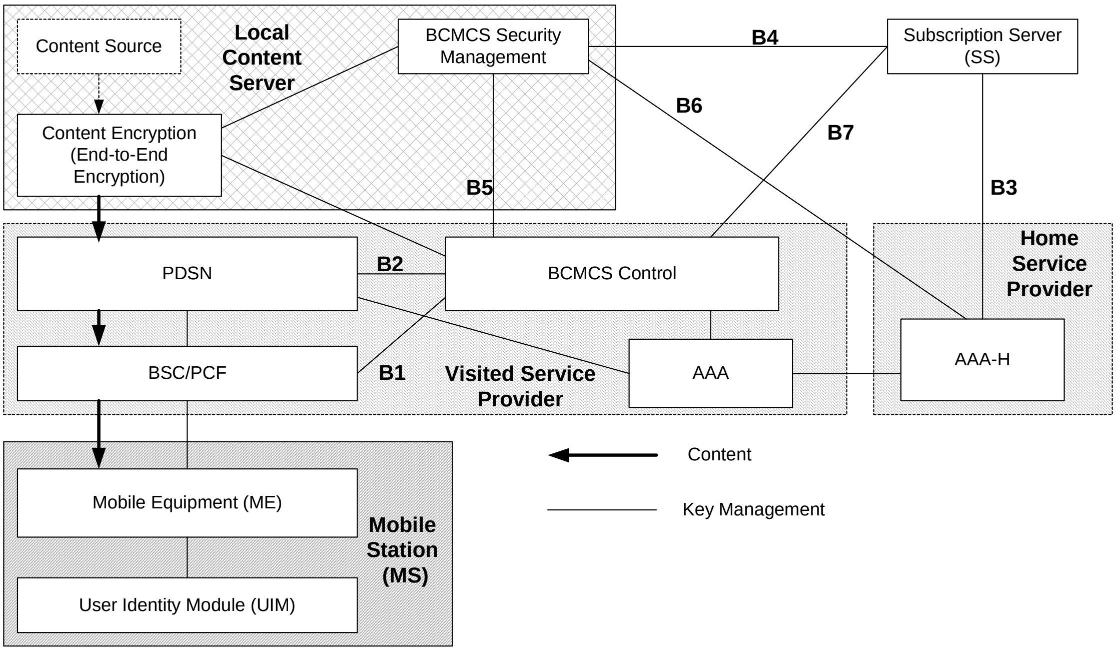


Figure 5.1. High-level Architecture

Figure 5.1 shows the entities that may be involved in BCMCS (if link-layer encryption is employed then the Content Encryption in the local CS is not required). The new interfaces are used for sending RK and BAK values between Service Providers and BCMCS entities. The functions (with respect to BCMCS) of the entities and new interfaces are described in more detail below.

5.2.1 Summary of Entities

<u>**HOME SERVICE PROVIDER (Home SP)**</u>

AAA-H Authentication, Authorization, and Accounting. The AAA-H currently holds voice/data subscriptions. The AAA-H shares a key K with the UIM: this key K may be the A-key (which is the basis of key distribution and authentication for voice/data services). *The provisioning of this key K in the UIM and AAA-H is beyond the scope of this document.* We shall consider the AAA-H as equivalent to the home service provider. The AAA-H may manage BCMCS subscriptions for its voice/data subscribers.

The AAA-H may also have business arrangements with third-party BCMCS Subscription Servers (SS), wherein the AAA-H will allow the SS's to offer BCMCS subscriptions to the AAA-H's voice/data subscribers. When a voice/data subscriber of the AAA-H first subscribes to a particular SS, the AAA-H establishes a Registration Key (RK) to be shared by the user's UIM and the SS.

VISITED SERVICE PROVIDER (Visited SP)

AAA: If the visited Service Provider is performing BAK management (BAK Encryption) for some BCMC services, and if a user uses a subscription held in the AAA-H, then the visited AAA can provide TK values to the BCMCS Control.

BCMCS Control: If the visited Service Provider is performing BAK management (BAK Encryption) for some BCMC services, then it is the BCMCS Control that performs this function.

BSC/PCF: With respect to security, the BSC/PCF is only involved in those BCMC services where link-layer encryption is applied. In this case, the BSC/PCF generates SK, encrypts SK and encrypts the content.

PDSN: Interfaces between the Internet and the RAN.

CONTENT SERVER

BCMCS Security Manager: This entity is required if the CS is performing its own BAK management. The Security Manager delivers BAK to the entity performing SK generation and content encryption Content Server's Content Encryption entity.

Content Encryption: Used when end-to-end encryption is applied. The Content Encryption entity also generates SK values.

Content Source: Generates the un-encrypted content.

BCMCS SUBSCRIPTION SERVER (SS): A third-party entity that holds the subscription data for the user. *Note. The SS may be part of the local CS. However, for the sake of simplicity, the SS will always be shown as a separate entity.*

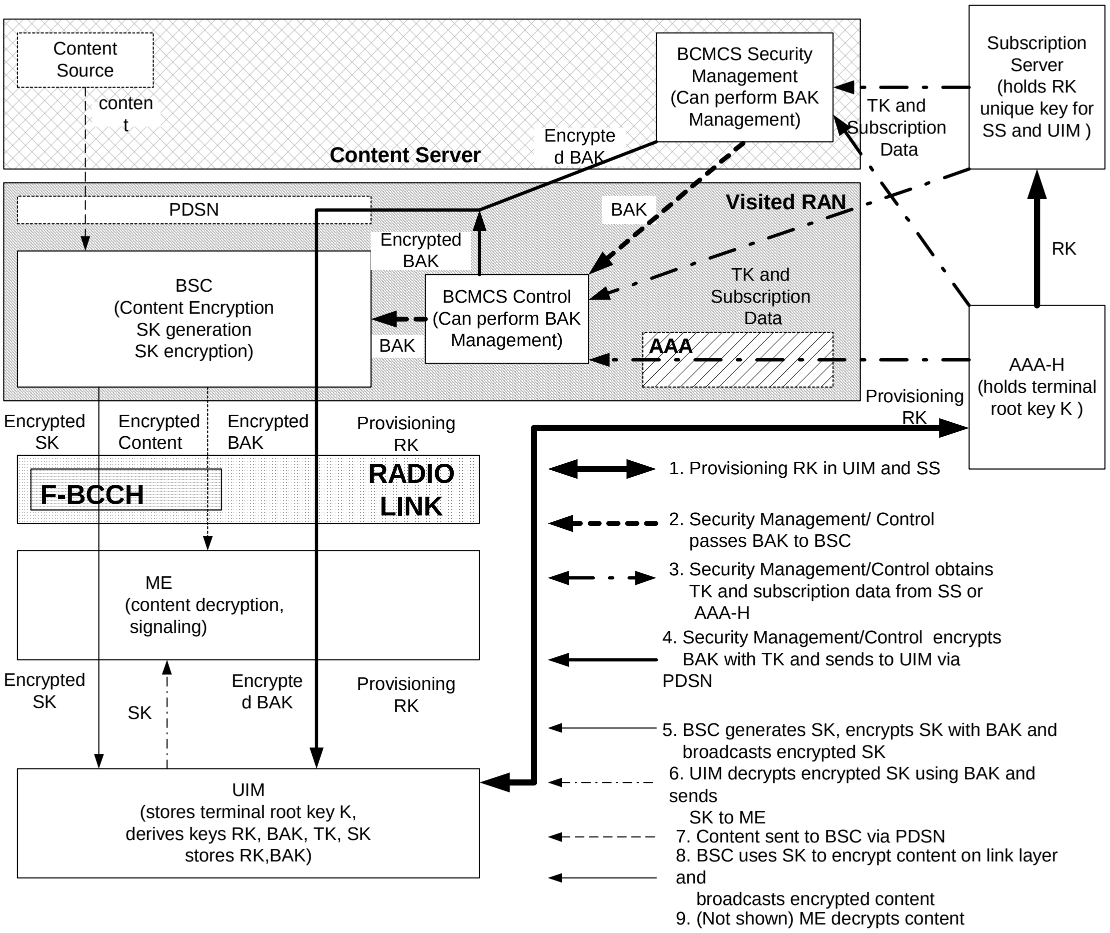
Mobile Station (MS) For the purposes of this document, the MS is considered as two separate entities, the UIM and ME.

UIM: User Identity Module (UIM): The UIM is a low power processor that contains secure memory. The UIM may be removable (like a SIM card) or part of the MS itself. The UIM shares a key K with the AAA-H. The UIM performs all key management related to BCMCS.

ME: Mobile Equipment. Includes equipment for receiving broadcast. The ME contains a high power processor, but no secure memory. The ME performs decryption.

5.2.2 Summary of Key Distribution for Link-Layer Encryption

Figure 5.2 shows the basic communications involved in the key distribution for BCMCS link-layer encryption. Many details are omitted in this diagram for the sake of clarity. The Figure is described below.



Content Source
conten t
Content Server
BCMCS Security Management (Can perform BAK Management)
Encrypte d BAK
TK and Subscription Data
Subscription Server (holds RK unique key for SS and UIM)
PDSN
BAK
Visited RAN
Encrypted BAK
BSC (Content Encryption SK generation SK encryption)
BCMCS Control (Can perform BAK Management)
BAK
TK and Subscription Data
AAA
RK
AAA-H (holds terminal root key K)
Provisioning RK
Encrypted SK
Encrypted Content
Encrypted BAK
Provisioning RK
F-BCCH
RADIO LINK
ME (content decryption, signaling)
Encrypted SK
SK
Encrypte d BAK
Provisioning RK
UIM (stores terminal root key K, derives keys RK, BAK, TK, SK stores RK,BAK)
1. Provisioning RK in UIM and SS
2. Security Management/ Control passes BAK to BSC
3. Security Management/Control obtains TK and subscription data from SS or AAA-H
4. Security Management/Control encrypts BAK with TK and sends to UIM via PDSN
5. BSC generates SK, encrypts SK with BAK and broadcasts encrypted SK
6. UIM decrypts encrypted SK using BAK and sends SK to ME
7. Content sent to BSC via PDSN
8. BSC uses SK to encrypt content on link layer and broadcasts encrypted content
9. (Not shown) ME decrypts content

Figure 5.2 High-level Architecture showing functional entities involved in BCMCS security when using link-layer encryption.

1. The UIM and SS agree on a Registration Key RK that will be the basis of authentication and key exchange with respect to BCMCS. The AAA-H assists in this process. *(If the AAA-H holds the subscription for the requested BCMCS, then the AAA-H may use K in the place of RK, and this step is not required).*

2. A BAK value allows subscribed users access for a period of time (the period of time is determined by the CS). The CS may manage BAK generation and encryption using the Security Manager, or the visited SP may manage BAK generation and encryption using the BCMCS Control. The Security Manager/Control generates the BAK values. The value of BAK, along with the

corresponding values of BAK_ID and BAK_lifetime, are passed to the BSC. The Security manager passes BAK to the BSC via the Control.

3. In order to send BAK to the UIM, the BAK must be encrypted. The Security Manager/Control obtains a temporary key (TK) generated by the subscription entity (SS or AAA-H). The subscription entity generates TK from a random value RAND_TK and RK . The subscription entity sends TK and RAND_TK to the Security Manager/Control. *(If the AAA-H holds the subscription then the AAA-H may use K in the palace of RK)*

4. The Security Manager/Control encrypts the value of BAK with TK, and sends the encrypted BAK, along with RAND_TK to the UIM via the PDSN. The UIM first forms TK from TK_RAND and RK, and then decrypts the encrypted BAK with TK to form BAK. The value of BAK is stored in the UIM.

5. The BSC generates an SK, encrypts SK with BAK, and broadcasts the value of the encrypted SK. If the user is listening to the BCMCS associated with this SK, then the MS passes the encrypted SK to the UIM.

6. The UIM obtains SK by decrypting the encrypted SK using the appropriate BAK. The UIM then passes the value of SK back to the broadcast receiver/ME.

7. The content is sent to the BSC via the PDSN.

8. The BSC encrypts the content using SK and has the visited SP broadcast the encrypted content.

9. (Not shown in Figure 4.3) The broadcast receiver/ME decrypts the content using SK. The broadcast receiver/ME is then able to process and display the content.

   - The process returns to step 5 when the BSC changes SK.

   - The process returns to Step 2 when the BAK Generator changes BAK.

5.2.3 Summary of Key Distribution for Link-Layer Encryption

Figure 5.3 shows the basic communications involved in the key distribution for BCMCS link-layer encryption. Many details are omitted in this diagram for the sake of clarity. The Figure is described below.

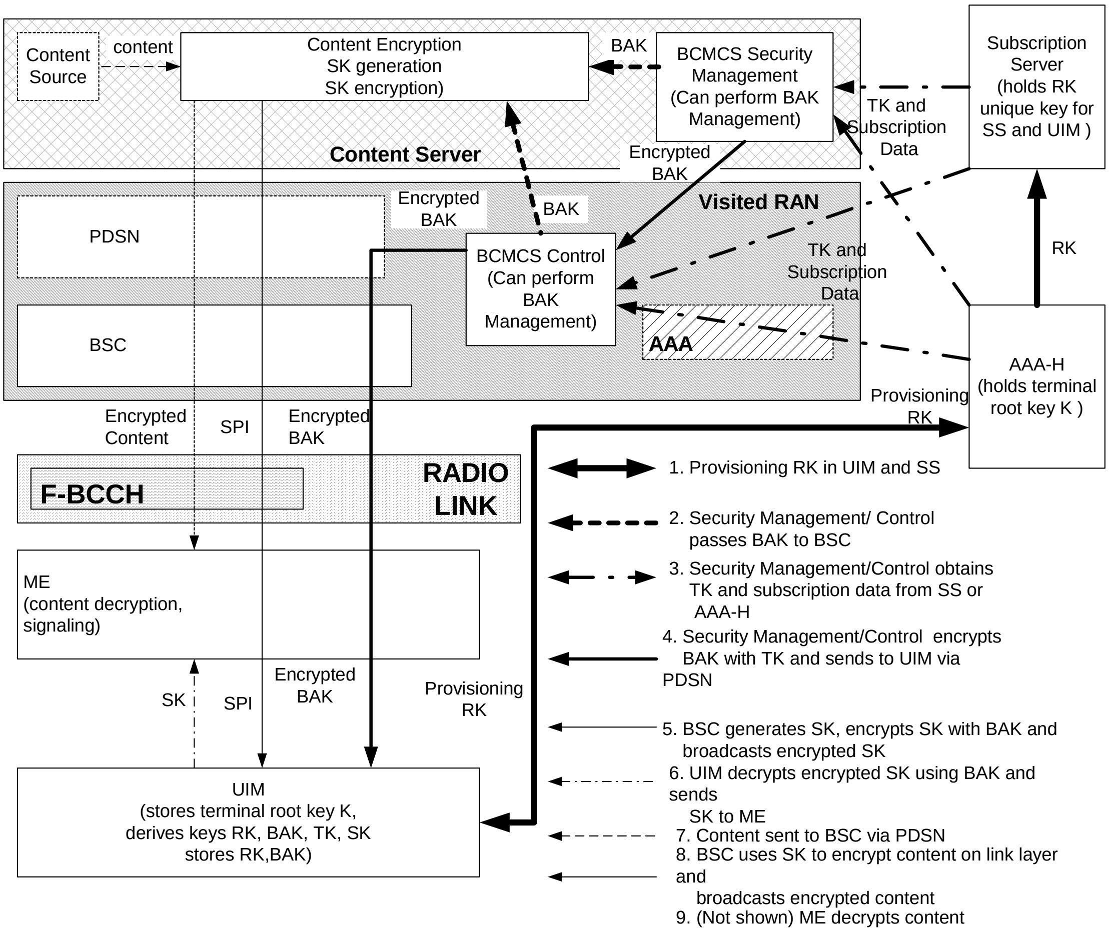

Content Source
content
Content Encryption
SK generation
SK encryption)
BAK
BCMCS Security Management (Can perform BAK Management)
Subscription Server (holds RK unique key for SS and UIM)
TK and Subscription Data
Content Server
Encrypted BAK
Encrypted BAK
BAK
Visited RAN
PDSN
BCMCS Control (Can perform BAK Management)
TK and Subscription Data
RK
BSC
AAA
AAA-H (holds terminal root key K)
Provisioning RK
Encrypted Content
SPI
Encrypted BAK
F-BCCH
RADIO LINK
1. Provisioning RK in UIM and SS
2. Security Management/ Control passes BAK to BSC
3. Security Management/Control obtains TK and subscription data from SS or AAA-H
ME (content decryption, signaling)
4. Security Management/Control encrypts BAK with TK and sends to UIM via PDSN
SK
SPI
Encrypted BAK
Provisioning RK
5. BSC generates SK, encrypts SK with BAK and broadcasts encrypted SK
6. UIM decrypts encrypted SK using BAK and sends SK to ME
UIM (stores terminal root key K, derives keys RK, BAK, TK, SK stores RK,BAK)
7. Content sent to BSC via PDSN
8. BSC uses SK to encrypt content on link layer and broadcasts encrypted content
9. (Not shown) ME decrypts content

Figure 5.2 High-level Architecture showing functional entities involved in BCMCS security when using link-layer encryption.

1. The UIM and SS agree on a Registration Key RK. *(AAA-H may use K in place of RK).*

2. The Security Manager/Control generates the BAK values. The value of BAK, along with the corresponding values of BAK_ID and BAK_lifetime, are passed to the BSC. The Security manager passes BAK to the BSC via the Control.

3. The Security Manager/Control obtains a temporary key (TK) generated by the subscription entity (SS or AAA-H). The subscription server generates TK from a random value TK_RAND, and sends TK and TK_RAND to the BCMCS Security Manager/Control. *(AAA-H may use K in place of RK).*

4. The Security Manager/Control encrypts the value of BAK with TK, and sends the encrypted BAK, along with TK_RAND to the UIM via the PDSN. The UIM first forms TK from TK_RAND and RK, and then decrypts the encrypted BAK with TK to form BAK. The value of BAK is stored in the UIM.

5. The BSC generates an SK, encrypts SK with BAK, and broadcasts the value of the encrypted SK. If the user is listening to the BCMCS associated with this SK, then the MS passes the encrypted SK to the UIM.

6. The UIM obtains SK by decrypting the encrypted SK using the appropriate TK. The UIM then passes the value of SK back to the broadcast receiver/ME.

7. The content is sent to the BSC via the PDSN.

8. The BSC encrypts the content using SK and has the visited SP broadcast the encrypted content.

9. (Not shown in Figure 4.3) The broadcast receiver/ME decrypts the content using SK. The broadcast receiver/ME is then able to process and display the content.

   - The process returns to step 5 when the BSC changes SK.

   - The process returns to Step 2 when the BAK Generator changes BAK.

5.3 RK Establishment

A *Registration Key* RK is a key shared only by the UIM and subscription entity. The RK is used to generate keys to securely send BAK values to the UIM without other entities being able to decrypt and obtain BAK. *The value of RK must be stored in secure memory in the UIM and must never be revealed.*

Each UIM will have a different RK for each subscription entity with whom the UIM has a subscription. Likewise, each subscription entity will have a different RK for each UIM subscribed through that subscription entity. If a UIM has different subscriptions with the same subscription entity, then the UIM may use the same RK for each subscription.

If the home service provider (that is, the AAA-H) holds the subscription, then there is no need to provision the UIM with an additional RK; the AAA-H and UIM are presumed to share some ket K, such as the A-key K. There is no harm in the AAA-H choosing to use a new RK for BCMCS, rather than choosing to use K, but in this case the AAA-H is logically equivalent to an SS.

The RK must be established before a user can subscribe to the service. However, if the SS or UIM chooses to change RK, then the UIM and SS simply repeat the RK establishment process shown in Figure 5.2 below.

If the subscription is held by the AAA-H, then the UIM can be pre-provisioned with RK. The UIM could also be pre-provisioned with RK values for “popular” third-party subscription servers. In these cases, the UIM should also be pre-provisioned with corresponding subscription identifiers (SUB_ID).

However, if the UIM is not pre-provisioned with an RK for a particular SS, then the UIM and SS establish RK when the user *first* begins the process of obtaining a subscription through SS. After RK has been established, the value of RK should not need to be changed again. The message flow for RK establishment is illustrated in Figure 5.2.

NEW FIGURE REQUIRED

Figure 5.2. The message flow for RK Establishment

The details of the steps for RK Establishment are as follows.

(a) The UIM and AAA-H are presumed to already share a key K. *The process of provisioning K is beyond the scope of this document.*

(b) The SS allocates the MS with a SUB_ID (including a SS_ID) and SS_IP_ADDR.

(c) The UIM or SS passes an RK request to the AAA-H.

(d) The SS provisions the UIM with an RK. *The process of provisioning the RK in the UIM is beyond scope of this document.*

(e) The AAA-H sends a message containing SUB_ID and RK to the SS.

(f) The SS stores the RK indexed by SUB_ID.

The UIM and SS now share a key RK. Only the UIM, SS and the AAA-H of the user know RK. The AAA-H of the user has a business relationship with the user and SS, so this means that the user and SS can trust the AAA-H; it is acceptable for AAA-H to know this key.

5.4 Subscription

There are many variations as to what a "subscription" provides. For example, a subscription may only provide access to a single broadcast services in the local network. Or a subscription may entitle the user to worldwide access to a variety of broadcast services. A user may also have a variety of subscriptions.

A user subscribes through either the user's home Service Provider or a BCMCS Subscription Server (SS). In the first case, subscription data is held in the AAA-H, while in the second case the SS holds the subscription data. Recall that the SS may be part of the local CS (although the diagrams will always show the SS as a separate entity).

The subscription process is outside the scope of this document. However, the subscription process should result in the UIM and subscription entity agreeing on a set of BCMCS identifiers (BCMCS_ID values) and BCMCS content identifiers (Content_ID values) to be associated with the user's subscription identifier SUB_ID. The SUB_ID was allocated by the subscription entity during RK establishment. The MS uses the SUB_ID when requesting BAK updates. This will tell the BCMCS Security Manager or BCMCS Control where to obtain subscription data and TK values.

5.5 BAK Management

A user can repeat the BAK update process with multiple BCMC services, and thereafter (from a security perspective) has the potential to be simultaneously accessing multiple BCMC services, from multiple Content Services.

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

BAK_ID. The BAK must be updated in the UIM prior to the BAK actually being used to encrypt/generate SK values (see Section 5.5.5). The UIM must have some means to determine which BAK is valid. The proposed method is to allocate a 4-bit BAK identifier BAK_ID to each BAK. When a BAK is provisioned, the UIM will index the BAK according to BCMCS_ID and BAK_ID. When SK values are encrypted using BAK (or generated using BAK), the BAK_ID will be included in the broadcast data. The UIM uses the BAK indexed by BCMCS_ID and BAK_ID. A BAK lifetime will also be provided, so the UIM can discard used keys.

When the MS determines that it needs to update BAK, the MS will be aware of the BAK_ID value corresponding to the BAK that the MS is requesting.

BAK lifetime. The CS decides how often BAK is changed. 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 billing. 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 CS 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.

In order to allow for various business models, two entities are provided that can manage BAK distribution: the BCMCS Control in the visited RAN or the BCMCS Security Manager in the CS corresponding to the requested BAK. If the CS does not wish to manage a Security Manager, then the CS can let the BCMCS Control manage the BAK distribution.

5.5.1 BAK Update by BCMCS Control in RAN

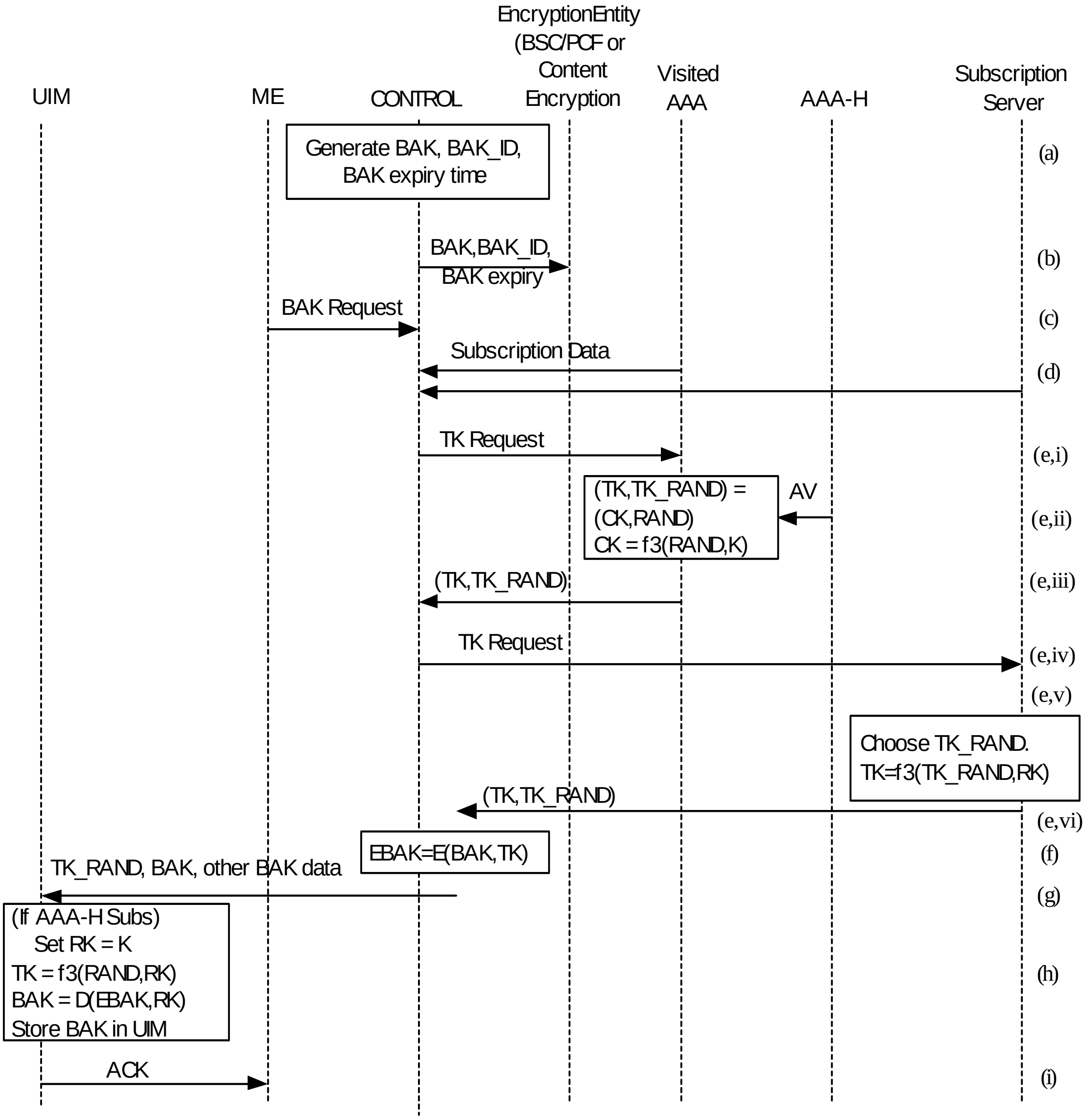

EncryptionEntity (BSC/PCF or Content Encryption
UIM
ME
CONTROL
Visited AAA
AAA-H
Subscription Server
Generate BAK, BAK_ID, BAK expiry time
(a)
BAK,BAK_ID, BAK expiry
(b)
BAK Request
(c)
Subscription Data
(d)
TK Request
(e,i)
(TK,TK_RAND) = (CK,RAND) CK = f3(RAND,K)
AV
(e,ii)
(TK,TK_RAND)
(e,iii)
TK Request
(e,iv)
(e,v)
Choose TK_RAND. TK=f3(TK_RAND,RK)
(TK,TK_RAND)
(e,vi)
EBAK=E(BAK,TK)
(f)
TK_RAND, BAK, other BAK data
(g)
(If AAA-H Subs) Set RK = K TK = f3(RAND,RK) BAK = D(EBAK,RK) Store BAK in UIM
(h)
ACK
(i)

Figure 5.3 BAK Update by the BCMCS Control

(a) The BCMCS Control generates BAK, BAK_ID and BAK expiry time.

(b) The BCMCS Control passes BAK, BAK_ID and BAK expiry time to either the BSC (if using link-layer encryption) or the CS Content Encryption (if using end-to-end encryption).

(c) The MS determines that it needs to update BAK and requests a BAK value from the network. The BAK request indicates BCMCS_ID (or Content_ID), BAK_ID, SUB_ID. The BAK request either includes a flag indicating that the subscription is held with the AAA_H or a third party SS. If the subscription is held with a third-party SS, then the BAK request includes SS_IP_ADDR the IP address of the SS. The BAK request is forwarded to the appropriate BCMCS Control via the PDSN.

(d) The BCMCS Control determines the subscription entity from SUB_ID and other flags. If the BCMCS Control does not have the current subscription details for the user, then the BCMCS Control obtains these details from the relevant subscription entity (AAA-H or third-party SS).

(e) The BCMCS Control needs a temporary key (TK, TK_RAND) pair to encrypt BAK. If the subscription is with the user's AAA-H, then

(e.i) The BCMCS Control sends a TK request to the visited AAA.

(e.ii) The visited AAA forms one or more (TK, TK_RAND) pairs corresponding to (CK, RAND) pairs extracted from Authentication Vectors produced by the user's AAA-H.

(e.iii) The AAA sends the (TK,TK_RAND) pairs to the BCMCS Control.

If the subscription is with a third-party SS, then:

(e.iv) The BCMCS Control sends a TK request to the SS, along with the SUB_ID.

(e.v) The SS chooses a random value RAND_TK and computes TK = ***f***(RAND_TK,RK) using some cryptographic function ***f***. In this way the SS generates one or more pairs (TK,RAND_TK).

(e.vi) The SS sends the pair(s) (TK, RAND_TK) to the BCMCS Control.

(f) The BCMCS Control forms EBAK by encrypting BAK using the key TK.

(g) The BCMCS Control sends EBAK, BAK_ID and other BAK data to the MS via the PDSN. The ME passes the encrypted BAK to the UIM.

(h) If the subscription is with the AAA-H, then UIM may use the A-key K for RK. The UIM forms TK = ***f***(TK_RAND,RK). The UIM forms BAK by decrypting EBAK using TK. The UIM stores BAK indexed by BAK_ID.

(i) The UIM sends an ACK to the ME, so that the ME knows that the BAK update was successful.

5.5.2 BAK Update by BCMCS Security Management in Content Server

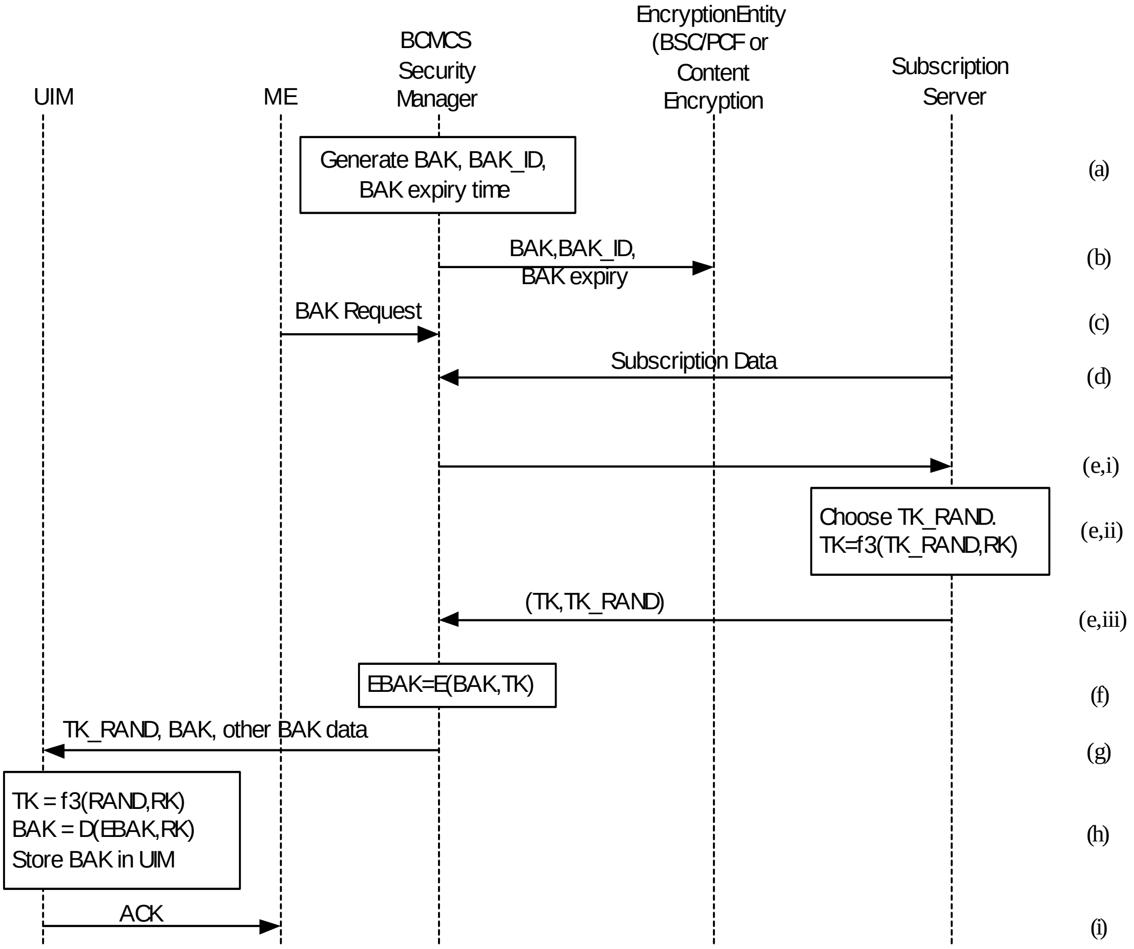


Figure 5.3 BAK Update by a BCMCS Security Manager (in the Content Server)

Most of the steps are identical to the previous section, with the Security Manager taking the place of the Control. However, unlike the BCMCS Control, the Security Manager cannot obtain “K-based” TK pairs from the visited AAA. This is because the AAA-H may not have a business agreement with the Security Manager, and so the AAA-H should not trust the Security Manager with TK values it has generated.

(a) The Security Manager generates BAK, BAK_ID and BAK expiry time.

(b) The Security Manager passes BAK, BAK_ID and BAK expiry time to either the BSC via the Control (if using link-layer encryption) or the CS Content Encryption (if using end-to-end encryption).

(c) The MS determines that it needs to update BAK and requests a BAK value from the Security Manager via the PDSN.

(d) If the Security Manager does not have the current subscription details for the user, then the Security Manager obtains these details from the relevant SS.

(e) The Security Manager needs a temporary key (TK, TK_RAND) pair to encrypt BAK. The Security Manager may re-use TK values. Whenever the Security Manager needs new TK values, the following sub-steps are performed.

(e.i) The Security Manager sends a TK request to the SS, along with the SUB_ID.

(e.ii) The SS chooses a random value RAND_TK and computes TK = ***f***(RAND_TK,RK) using some cryptographic function ***f***. In this way the SS generates one or more pairs (TK,RAND_TK).

(e.iii) The SS sends the pair(s) (TK, RAND_TK) to the Security Manager.

(f) The Security Manager forms EBAK by encrypting BAK using the key TK.

(g) The Security Manager sends EBAK, BAK_ID and other BAK data to the MS via the PDSN. The ME passes the encrypted BAK to the UIM.

(h) The UIM forms TK = ***f***(TK_RAND,RK). Then the UIM forms BAK by decrypting EBAK using TK. The UIM stores BAK indexed by BAK_ID.

(i) The UIM sends an acknowledgement message ACK to the ME, so that the ME knows that the BAK update was successful.

5.5.3 Authenticating BAK Requests

An adversary achieves nothing 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 BCMCS Security Manager /Control does not need to authenticate BAK requests.

5.5.4 Storage of BAK

It is IMPERATIVE that the UIM does not reveal BAK. If a single UIM reveals BAK, then all security is lost until the CS changes BAK.

The UIM should store BAK and related data about BAK, such as BAK_ID and expiration time, if any. The BAK is indexed by the BCMCS_ID (of the corresponding BCMCS) and BAK_ID. The UIM discards any BAK values that have expired.

The ME may store the BAK-related data, to save requesting this information from the UIM.

5.5.5 Updating BAK before use

It may prove beneficial to provision UIM with BAK shortly before BAK begins being used to derive SK values. Otherwise, once the CS starts sending packets with SK derived from the new BAK, the user would experience a delay as the MS 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, BCMCS may allow an MS to obtain the new BAK shortly before the BAK changes. The MS, BCMCS Security Manager /Control or CS may initiate this process. Different MS 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.

5.6 Encryption/Decryption

5.6.1 SK Lifetime

The same value of SK can be used for more than one IPSec packet or radio frame. The CS decides how often to change SK. Decreasing the lifetime of SK increases the security. We recommend that the SK should normally change every 5 to 10 minutes, but it is up to the CS to decide how often they wish to change SK. During peak-usage times, the CS may choose to change SK more often for additional security.

5.6.2 Storage of SK

The UIM passes SK to the ME, and the ME stores SK in the security association indexed by the value of SPI.

5.6.3 Link-Layer Encryption

When distributing SK values, a problem arises in ensuring that a listening MS will change SK at the right time. There is the possibility that packets may arrive in incorrect order, so the MS receives the new SK either too early, or too late. When the BSC/PCF is performing the encryption, this is not a problem because the BSC/PCF can send the new SK and then immediately start using this new SK: the packets are guaranteed to arrive in the correct order.

5.6.3.1 SK Generation and SK Encryption

If the BSC has already computed the SK, then the CS encrypts the broadcast content according to IPSec ESP in transport mode, using SK as the encryption key.

1. The BSC/PCF chooses a random 128-bit value for SK.

2. The BSC/PCF forms ESK by applying some encryption to the 128-bit value SK using BAK as the key.

3. The BSC/PCF regularly broadcasts (BCMCS_ID, BAK_ID, SK_ID, ESK) where BCMCS_ID identifies the BCMC service, BAK_ID identifies the current BAK value and SK_ID identifies the SK. *The SK_ID is used by the ME to determine when SK changes, without needing to decrypt ESK. A simple counter should suffice.*

4. The BSC/PCF can use the new SK as soon as it broadcasts the first (BCMCS_ID, BAK_ID, SK_ID, ESK) formed from the new SK.

After a user begins receiving a BCMC service, there may be a delay before the MS receives the broadcast value of ESK (the encrypted value of SK). The ESK should be broadcast frequently, so as to minimize this delay.

5.6.3.2 SK Decryption

1. The ME receives (BCMCS_ID,BAK_ID,SK_ID,ESK) on the broadcast channel. If the received value of SK_ID is different from the SK_ID value for the current value of SK, then this indicates that ESK is the encryption of a new key.

2. The ME passes BCMCS_ID, BAK_ID and ESK to the UIM.

3. The UIM forms SK by decrypting ESK using the BAK indexed by BCMCS_ID and BAK_ID.

4. The UIM passes SK back to the ME

5. The ME resets the decryption key to the value of SK returned by the UIM. The ME also stores SK_ID. The ME begins using this decryption key SK immediately.

5.6.3.3 Content Encryption and Content Decryption

Presume that the BSC/PCF has generated, encrypted and broadcast SK, and the ME has determined SK.

1. The BSC/PCF broadcasts the content, applying link-layer encryption using the key SK as the cipher key.

2. The ME decrypts the broadcast with link-layer encryption using the key SK as the cipher key.

5.6.4 End-to-End IPSec Encryption

When distributing SK values, a problem arises in ensuring that a listening MS will change SK at the right time. To resolve this problem for IPSec encryption, the value of SK is linked to the SPI value in the IPSec header. In particular, thee value of SK is deermined directly from SPI and BAK. This also means that the MS gets the decryption keys with the encrypted content, rather than SK being sent separately.

5.6.4.1 SK Generation and Content Encryption

If the CS Content Encryption has already computed the SK corresponding to the current SPI, then the Content Encryption encrypts the broadcast content according to IPSec ESP in transport mode, using SK as the encryption key.

1. The Content Encryption chooses a random 28-bit value SPI_RAND.

2. The Content Encryption forms a 32-bit SPI of the form SPI = (BAK_ID ||SPI_RAND), where the 4-bit BAK_ID identifies the current BAK value.

3. The Content Encryption pads SPI_RAND to 128 bits.

4. The Content Encryption encrypts this 128-bit value using BAK as the key. The 128-bit output is SK.

5. The Content Encryption puts the new value of SK in an SA indexed by the SPI and the destination address of the broadcast packet.

The value of SPI_RAND should be random, so that a user can't predict what values of SPI will be used in the future. Otherwise someone could pre-compute the SK values to be used for that day and distribute these keys at the beginning of the day. For someone wanting to distribute keys, this process is easier (and less expensive) [1] than distributing the keys in real time.

5.6.4.2 SK Generation and Content Decryption

Given a BCMCS IPSec packet, the ME performs the following steps.

1. The ME obtains the SPI and BCMCS_ID. The ME extracts the BAK_ID from the SPI. The ME then decides if the UIM has the correct BAK. If the UIM doe not have the correct BAK, then the MS updates the BAK as discussed above. (Alternatively, the ME may check the SDP to see if it has the current BAK, as discussed above).

2. The ME checks if it has a security association (SA) corresponding to the SPI and the destination address (or BCMCS_ID) of the broadcast packet. If the ME has an SA with this SPI and BCMCS_ID, then the ME decrypts the block (as per IPSec ESP in transport mode) using the decryption key SK in the SA.

3. If the ME does not have an SA with this SPI and BCMCS_ID, then the ME passes the SPI to the UIM so that the UIM can compute the SK.

4. The UIM computes the SK as follows.

[1] If a subscriber can pre-compute the keys for the day, then they could distribute the keys for free from his PC, and his "friends" can download for free from their PC. If the keys can't be predicted, then the "friends" would require frequent downloads over the wireless network: much more expensive.

a. The UIM extracts the BAK_ID from the 4 most significant bits of the SPI and retrieves the value of BAK (corresponding to BCMCS_ID and BAK_ID) from its memory.

b. The UIM extracts the 28-bit SPI_RAND and pads SPI_RAND to 128 bits.

c. The UIM encrypts this 128-bit value using BAK as the key. The 128-bit output is SK.

5. The UIM passes SK to the ME.

6. The ME puts the new value of SK in an SA indexed by the SPI and the destination address of the broadcast packet.

7. The ME now decrypts the block, as per IPSec ESP in transport mode, using SK as the key.

5.7 Entities, Functionality and Interfaces

HOME SERVICE PROVIDER

AAA-H Authentication, Authorization, and Accounting. The AAA-H currently holds voice/data subscriptions. The AAA-H shares a key K with the UIM: this key K is the basis of key distribution and authentication for voice/data services. We shall consider the AAA-H as equivalent to the home service provider. For the AAA-H to be able to manage BCMCS subscriptions for its voice/data subscribers, the AAA-H must have the following (additional) functionality.

- The AAA-H provides BCMCS subscription services for the AAA-H's voice/data subscribers. That is, the AAA-H provides the mechanism for the user to subscribe to (and un-subscribe from) BCMC services. The subscription data for BCMCS is included in the user profile.

- The AAA-H provides billing information (related to BCMCS) to the BCMCS Control.

The AAA-H may also have business arrangements with third-party BCMCS Subscription Servers (SS), wherein the AAA-H will allow the SS's to offer BCMCS subscriptions to the AAA-H's voice/data subscribers. For the AAA-H to offer such arrangements, the AAA-H must have the following functionality.

- When a voice/data subscriber of the AAA-H first subscribes to a particular third-party SS, the AAA-H establishes a Registration Key (RK) to be shared by the user's UIM and the SS.

VISITED SERVICE PROVIDER (Visited SP)

AAA: The roles of the visited AAA with respect to BCMCS are as follows.

- The visited AAA interacts with the UIM and AAA-H during AKA procedures involved in RK establishment.
- If a user uses a subscription held in the AAA-H, then the visited AAA provides temporary keys to the BCMCS Control.
- The AAA provides accounting information to the Content Servers that broadcast content on that RAN.
- Entities outside the RAN that wish to communicate with entities within the RAN may communicate via the AAA.

BCMCS Control: The BCMCS must have the following functionality.

- The BCMCS Control creates the following association for flexible BCMCS_ID local assignment
    - (Content Provider ID, Content ID) (universal),
    - BCMCS_ID (non-universal),
    - (Multicast IP address, Port number) (non-universal).

  See Section 0 for definitions of Content_ID and BCMCS_ID.
- The BCMCS Control performs BCMCS content flow treatment.
- For each BCMCS, the BCMCS Control informs the MS and BSC about which layer encryption is performed at (Link Layer, IP Layer). The BCMCS Control obtains this information from the local Content Servers via out of band means.
- The BCMCS Control informs the transport and application protocol.
- If link layer encryption is used, then
    - The BCMCS Control passes BAK values to the BSC/PCF so that the BSC/PCF can encrypt SK values.
    - The BCMCS Control passes the SK lifetime values to the BSC/PCF.
- If the visited Service Provider is performing BAK management (BAK Encryption) for some BCMC services then the BCMCS Control:

    - Generates BAK values for these BCMC services.
    - Obtains
    - Obtains TK values from:
        - The AAA-H via the visited AAA,
        - The AAA-H directly, or
        - An SS directly.
    - Encrypts BAK using TK.
- The BCMCS Control downloads encrypted BAK values to the UIM of BCMCS-subscribed users via the PDSN.

BSC/PCF:

- The BSC/PCF performs other functions with relation to BCMCS that are not relevant to this document. For example, the BSC/PCF receives BCMCS registration messages from mobile stations. These messages are used to determine the number of user's accessing the content, and to determine what channel the MS should be paged on. The BSC/PCF determines what BCMC services will be transmitted on the radio link. The BSC/PCF also generates and transmits the necessary overhead messages.
- For each BCMC service that requires link-layer encryption:
    - The BSC/PCF generates SK.
    - The BSC/PCF encrypts SK with the current BAK for that BCMC service.
    - The BSC/PCF delivers the encrypted SK to the UIM.
    - The BSC/PCF encrypts the content on the link-layer using the key SK.
    - Periodically, the BSC generates a new SK, based on the SK lifetime provided by the BCMCS Control.
- The BSC/PCF broadcasts the content (which may or may not be encrypted).

PDSN: Interfaces between the Internet and the RAN.

- The PDSN delivers the broadcast content to the MS.

- Encrypted BAK values are transmitted via the PDSN.

- The PDSN records accounting information.

- The PDSN establishes individual security associations with each Content Server that sends broadcast content through the PDSN. The security associations allow the Content Server to apply IPSec Authentication Header in tunnel mode. The PDSN verifies the Authentication Headers on the broadcast content IP packets it receives. In this way the PDSN authenticates the content on behalf of the user's receiving the broadcast.

CONTENT SERVER

BCMCS Security Manager: This entity is required if the CS is performing its own BAK management. The Security Manager has the following functions.

- The Security Manager generates BAK

- The Security Manager obtains TK values.

  - If the subscription is held by the user's AAA-H, then the Security Manager can obtain TK values from the visited BCMCS Control. The Control obtains the temporary keys from the visited AAA (who extracts the TK values from Authentication Vectors).

  - If a third-party SS holds the subscription, then the Security Manager obtains the TK values directly from this Subscription Server.

- The Security Manager downloads the encrypted BAK to the UIM via the visited BCMCS Control and PDSN.

- If end-to-end encryption is used, then

  - The Security Manager delivers BAK to the Content Server's Content Encryption entity.

  - The Security Manager determines the SK lifetime and sends this to the Content Server's Content Encryption entity.

- If link-layer encryption is used, then

  - The Security Manager delivers BAK to the BSC via the visited BCMCS Control.

    - The Security Manager determines the SK lifetime and sends this to the visited BCMCS Control.

Content Encryption: Used when end-to-end encryption is applied. The Content Encryption entity has the following functions:

- Receive content from a Content Source.
- Receive BAK, BAK_ID, BAK expiry time and SK lifetime from the Control or Security Manager.
- Generate short-term key (SK) values.
- Using the SK values, encrypt the content packets according to IPSec Encapsulated Security Payload (ESP) in transport mode.
- Send the IPSec packets to the PDSN for broadcast.

Content Source: Generates the un-encrypted content.

BCMCS SUBSCRIPTION SERVER (SS): A third-party entity that holds the subscription data for the user. An SS has the following functionality.

- The SS provides subscription for BCMC services to users that have voice/data subscriptions. The SS determines the mechanism for the user to subscribe to (and un-subscribe from) BCMC services. The SS stores the subscription data.
- The SS provides billing information to the Control (in visited SP) or Security Manager (in CS) as required.
- The SS provides subscription data to the Control (in visited SP) or Security Manager (in CS) as required.
- The SS obtains RK values for the AAA-H as part of RK Establishment.
- The SS generates Temporary Key (TK) values.
- The SS provides TK values to Control or Security Manager as required.
- The SS may contact Control entities or Security Managers to revoke subscriptions.

Note. The SS may be part of the local CS. However, for the sake of simplicity, the SS will always be shown as a separate entity.

5.7.1 Interfaces

B1 Interface (BCMCS Control – BSC/PCF)

- The B1 interface is used for various control functions not related to security.
- If link-layer encryption is used, then
    - The B1 interface delivers BAK to the BSC for encrypting SK.
    - The B1 interface delivers SK lifetime to the BSC.

B2 Interface (BCMCS Control – MS)

- The B1 interface is used for various control functions not related to security.
- The B1 interface is used to download the encrypted BAK to the UIM via PDSN.
- The B1 interface is used to tell the MS which layer performs encryption.

B3 Interface (AAA-H – Subscription Server)

- The B3 interface delivers RK to the Subscription Server from AAA-H

B4 Interface (Security Manager – Subscription Server)
(Security Manager – AAA-H)
(Control – Subscription Server)
(Control – AAA-H)

- The B4 interface delivers TK values to local BCMCS Security Manager or BCMCS Control.
- The B4 interface delivers subscription data to local BCMCS Security Manager or BCMCS Control.

B5 Interface (Local BCMCS Security Manager –BCMCS Control)

- If the Security Management is performing BAK Management, then B5 delivers the encrypted BAK values (generated by the Security Manager) to the Control.
- If link-layer encryption is used, then
    - The B5 interface sends SK lifetime to BCMCS Control
    - The B5 interface delivers BAK values to BCMCS Control

B6 Interface (Security Manager – AAA-H)

- The B6 interface delivers TK values to local BCMCS Security Manager.

- The B6 interface delivers subscription data to local BCMCS Security Manager.

B7 Interface (Control – Subscription Server)

- The B7 interface delivers TK values to visited BCMCS Control.

- The B7 interface delivers subscription data to visited BCMCS Control.

Interface between BCMCS Control and Content Server

- Beyond the scope of the standard

- The CS must indicate as to whether IP layer encryption, link-layer encryption or no encryption is used.

- If IP layer encryption is used, and the BCMCS Control is performing BAK management, then this interface is used for sending BAK to the Content Encryption in the Content Server.

Interface between the visited AAA and BCMCS Control (in the same RAN)

- Beyond the scope of the standard

- If a user’s BCMCS subscription is held in the user’s AAA-H, then the visited AAA passes TK values and subscription data from the AAA-H to the BCMCS Control.

5.8 Tables

Table 5.1 may be useful as a quick reference regarding the use, computation and storage of keys in the MS. Table 5.2 may be useful as a quick reference regarding the use, computation and storage of keys in the network.

The key:	is used to:	lasts for:	requires:	is stored in:
RK	Generate TK	[Permanent]	Root Key K	(temporary)
TK	Encrypt BAK	(temporary)	RK (or K)	(temporary)
BAK	Encrypt SK or Generate SK	hours/days	TK	UIM
SK	Decrypt content	sec/minutes	BAK and ESK or SPI	ME

Table 5.1. A summary of the use, computation and storage of keys in the Mobile Station. All keys are generated in the UIM.

The key:	generated by	where it is	This key is passed to	where it is used to:	is stored in:
RK	AAA-H	-	SS	generate TK	SS
TK	AAA-H or SS	-	Security Manager or Control	encrypt BAK	Security Manager or Control
BAK	Security Manager or Control	encrypted and passed to UIM	CS Content Encryption or BSC/PCF	Generate or encrypt SK.	CS Content Encryption or BSC/PCF
SK	CS	used to encrypt content.	-	-	CS Content Encryption or BSC/PCF

Table 5.2. A summary of the use, computation and storage of keys for the CS, SS and RAN entities.

6 Motivation

6.1 Design Philosophy

The major threat addressed in this document is that of user(s) accessing the broadcast content without paying the fees. This threat applies only when access is controlled on a subscription basis.

6.1.1 A Specific Goal

To access the broadcast content, a user must have the current decryption keys. The UIM 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. Eventually, someone will work out how to extract the current decryption key from the ME. A subscribed user will then be able to distribute the decryption key to other non-subscribed users. So it will be 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. So (with trepidation) I venture to suggest that the market is for

- Users that are mobile.

- Users that want quality service that is easy to access.

The real 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.

6.1.2 The Solution

Our solution was 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 broadcast. The BAK is stored in the UIM.

The ME cannot be trusted to store or compute long-term keys: these must be stored and computed in the UIM. The UIM is not powerful enough to perform public-key cryptographic operations so all key management must be based on symmetric cryptography.[1]

6.2 Motivation for Establishing Registration Keys

Recall that the purpose of the BCMCS 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 BCMCS Security Manager or BCMCS Control by providing the SUB_ID of a subscribed UIM. 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 BCMCS Security Manager /Control, and obtain BAK.

There are two ways to counter this attack.

1. The BCMCS Security Manager or BCMCS Control could get the local SP to authenticate each BAK request to ensure that the UIM has used the correct SUB_ID. This is expensive.

2. The BAK could be encrypted using a key known only to the correct UIM (that is, the UIM that corresponds to the SUB_ID). In this case, other UIM's will not be able to decrypt the packets sent back by the BCMCS Security Manager /Control. This provides implicit authentication of the UIM.

[1] This statement ignores the NTRU cryptosystem. The cryptographic community is not yet convinced of the security of the NTRU cryptosystem.

The second option appears the better choice. An adversary achieves nothing 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 BCMCS Security Manager /Control does not need to authenticate BAK requests.

6.3 Motivation for SK Management Design

A user may "tune in" at any time during the broadcast. The user will want almost instant access to the content. Thus, if information for deriving SK (for example, and encrypted value of SK or a random seed) is sent in a packets separate from the content, then the CS must resend the information for deriving SK every few seconds. One disadvantage is that this method uses up bandwidth. The major disadvantage is that there is no standard method for distinguishing packets containing SK information from packets containing content.

6.3.1 Link-Layer Encryption

(YET TO EXPLAIN)

6.3.2 End-to-End IPSec Encryption

We considered many options for distributing SK. In the normal instance of using IPSec, the two parties would normally negotiate when changing keys. Once the parties agree on the new keys, the SPI does not change: the parties simply place the new keys in the old security association and leave the SPI as it was. In BCMCS, there is a different situation because there are multiple receivers and the communication flows only from the CS to the users. The CS is not in a position to verify that the users have the correct value of SK. Similarly; the users have difficulty verifying that they have the correct value of SK. We found that changing the SPI when SK changes best solves this problem. This way the CS knows that the users are aware that SK has changed. We know that this is not standard practice in IPSec, but we are convinced that this is unavoidable.

The two major options for distributing SK were to either send SK in packets separate from the content stream, or to derive SK from information included in the IPSec packet containing the content. Hybrid schemes were also considered.

Given that the SPI is changing when SK is changing, we decided to take the addition step of deriving SK exclusively from BAK and the SPI. To ensure that the correct value of BAK is used, the SPI includes a 4-bit BAK_ID, and there would be an expiration time for the BAK so that BAK_ID can be re-used for other values of BAK in the future. This left 28 bits of the SPI that could change, corresponding to 2^{28} possible values of SK. When the ME comes across a new SPI, the ME passes this SPI to the UIM and the UIM computes SK from SPI and BAK. The ME would have the new SK back in a negligible amount of time, and could continue decrypting. The variable portion of SPI should be random; otherwise a subscribed user could get the UIM to pre-compute the necessary SK values and distribute them.

This method has several advantages:

1. There is no bandwidth required for distributing SK to the users.

2. This method allows the UIM to compute SK as soon as it has the BAK and the ME has begun receiving the IPSec packets. The user doesn't have to wait for the packets containing information for SK. This is a considerable advantage, particularly in the case where a user is changing channels every few seconds or minutes: the user will not want a few seconds delay while waiting for information to derive SK every time they change channels.

The main disadvantage to this scheme is the relatively small number of SK values that can be derived from a single BAK: 2^{28} values (corresponding to the 2^{28} values of SPI_RAND), as compared to 2^{128} values using other methods. It has been suggested that group of subscribed users could get their UIM to pre-compute all 2^{28} values of SK for the current BAK by inputting all 2^{28} possible SPI values. We estimate that one UIM might be able compute all the keys in around 3 days. You would need a large number of subscribed UIMs to be able to pre-compute these values within, say, one hour. This group could then distribute these values. The set of keys would require around 4 gigabytes of memory. However, since we are only concerned with users accessing via a PDA or phone, it is highly unlikely that they will have access to sufficient storage for all 4 gigabytes (beside this fact, the user will probably be unwilling to download 4 gigabytes every time that the BAK changes). Also, we are only concerned with users that want quality service, and without all the keys, the users are not going to be able to decrypt all the content, and won't get quality service. So such a scheme is unlikely to be a concern for BCMCS: 2^{28} SK

Update on BCMCS Security Framework

Phil Hawkes

QUALCOMM Australia

phawkes@qualcomm.com

Outline

- BCMCS Security
  - Entities
  - Key Hierachy
  - High-level Architecture
- Some proposed options
  - Encryption layer
  - BAK Update
  - Location of BAK encryptor
  - SK transport
- Key verification

Subscription

- Content Server/Provider/Source may not be the entity with whom the user subscribes:
  - Mainly for roaming.
  - E.g. CNN may have a central subscription entity. CNN Asia may be generated locally. CNN_Asia may have to check with CNN subscription database to authorize
- Presume separate *Subscription Server* (*SS*)
  - Complicates authentication and key distribution
  - Owned by local CS <u>or</u> has business agreement with local CS

Logical Entities

- **HLR/AC:** holds mobile phone subscription
- **MSC/VLR:** visited system/network: maybe VLR=HLR if user is not roaming
- **Content Server (CS):** A local/visited CS is one that provides content to the visited network (where user is located). A CS should include
  - Content source (may be from other sources)
  - BAK Generator
- **BAK Encryptor:** encrypts BAK to provision into UIM
  - May be associated only with CS or may serve many CS's
- **Subscription Server (SS)**
  - holds subscription data authorizing user to some BCMCSs
  - Owned by local CS <u>or</u> has business agreement with local CS

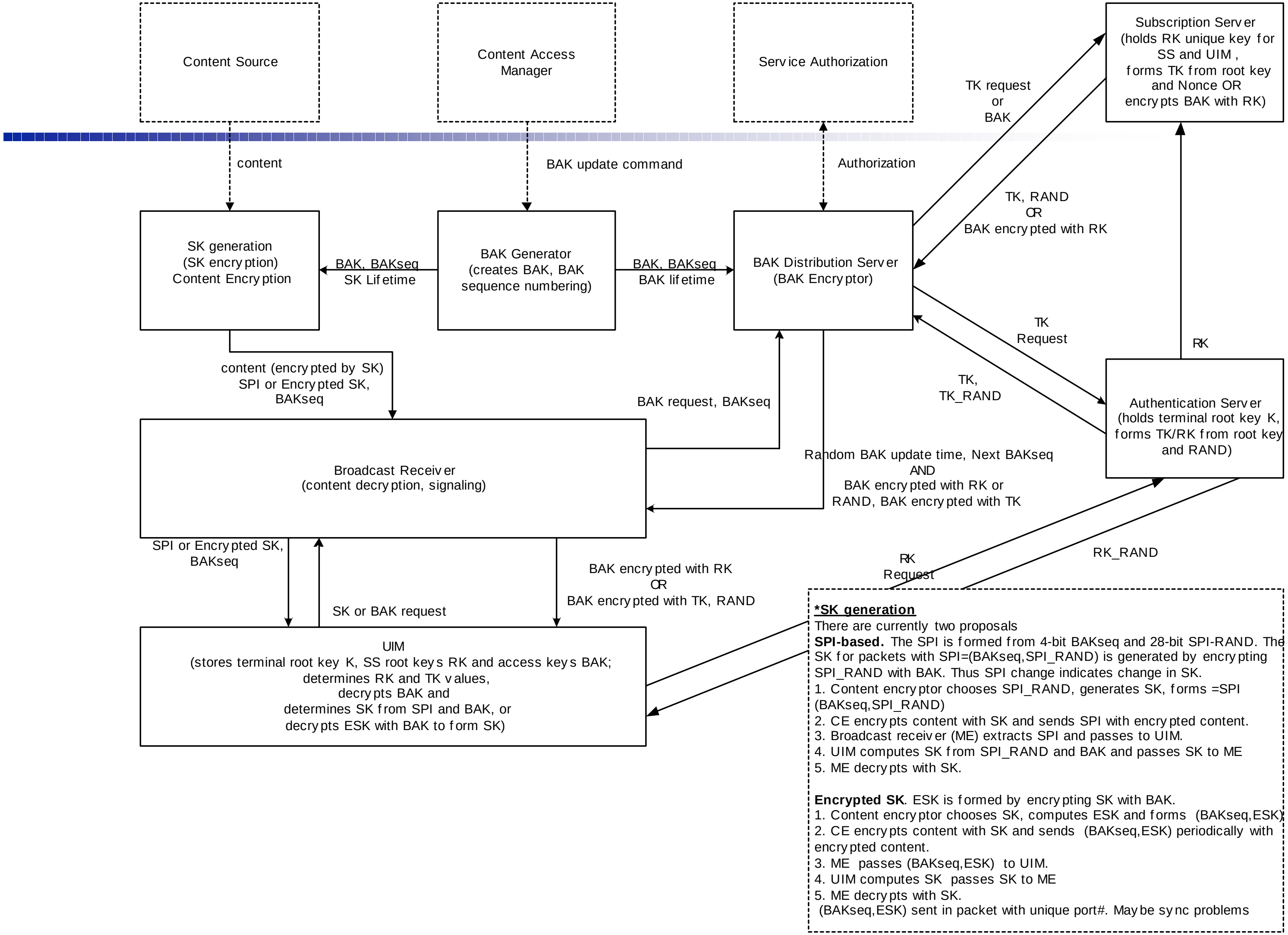
BCMCS Key Hierarchy

- **K=A-key: basis of AKA, held in HLR/AC and UIM**
- **RK (Root/Registration Key)**
  - **Held in Subscription Server (SS) and UIM**
    - **generating keys / authenticating UIM**
- **TK (Temporary Key)**
  - **Used by BAK Encryptor to encrypt BAK values**
    - **Generated by SS**
    - **Held in BAK Encryptor and UIM**
- **BAK (Broadcast Access Key)**
  - **Multiple decryption keys (SK) derived from single BAK**
    - **Held in BAK Encryptor and UIM**
- **SK (Short-term Key)** ***used to encrypt content***
    - **Derived in UIM using BAK and passed to ME**
    - **Changes frequently**

Key Hierarchy (more details)

- K=A-key
  - Root key for generating keys / authenticating UIM for normal mobile service
- RK (Root/Registration Key)
  - Derived from K
  - Permanent
- TK (Temporary Key)
  - If generated by HLR/AC then based on K
  - If generated by SS then based on RK
  - May be re-used
- BAK (Broadcast Access Key)
  - Multiple decryption keys (SK) derived from single BAK
  - BAK allows access until value is changed

HIGH LEVEL ARCHITECTURE (SECURITY)


Content Source
Content Access Manager
Service Authorization
Subscription Server (holds RK unique key for SS and UIM , forms TK from root key and Nonce OR encrypts BAK with RK)
TK request or BAK
content
BAK update command
Authorization
TK, RAND OR BAK encrypted with RK
SK generation (SK encryption) Content Encryption
BAK, BAKseq SK Lifetime
BAK Generator (creates BAK, BAK sequence numbering)
BAK, BAKseq BAK lifetime
BAK Distribution Server (BAK Encryptor)
TK Request
RK
TK, TK_RAND
content (encrypted by SK) SPI or Encrypted SK, BAKseq
BAK request, BAKseq
Authentication Server (holds terminal root key K, forms TK/RK from root key and RAND)
Broadcast Receiver (content decryption, signaling)
Random BAK update time, Next BAKseq AND BAK encrypted with RK or RAND, BAK encrypted with TK
SPI or Encrypted SK, BAKseq
BAK encrypted with RK OR BAK encrypted with TK, RAND
RK Request
RK_RAND
SK or BAK request
UIM (stores terminal root key K, SS root keys RK and access keys BAK; determines RK and TK values, decrypts BAK and determines SK from SPI and BAK, or decrypts ESK with BAK to form SK)

***SK generation**

There are currently two proposals

SPI-based. The SPI is formed from 4-bit BAKseq and 28-bit SPI-RAND. The SK for packets with SPI=(BAKseq,SPI_RAND) is generated by encrypting SPI_RAND with BAK. Thus SPI change indicates change in SK.

1. Content encryptor chooses SPI_RAND, generates SK, forms =SPI (BAKseq,SPI_RAND)
2. CE encrypts content with SK and sends SPI with encrypted content.
3. Broadcast receiver (ME) extracts SPI and passes to UIM.
4. UIM computes SK from SPI_RAND and BAK and passes SK to ME
5. ME decrypts with SK.

Encrypted SK. ESK is formed by encrypting SK with BAK.

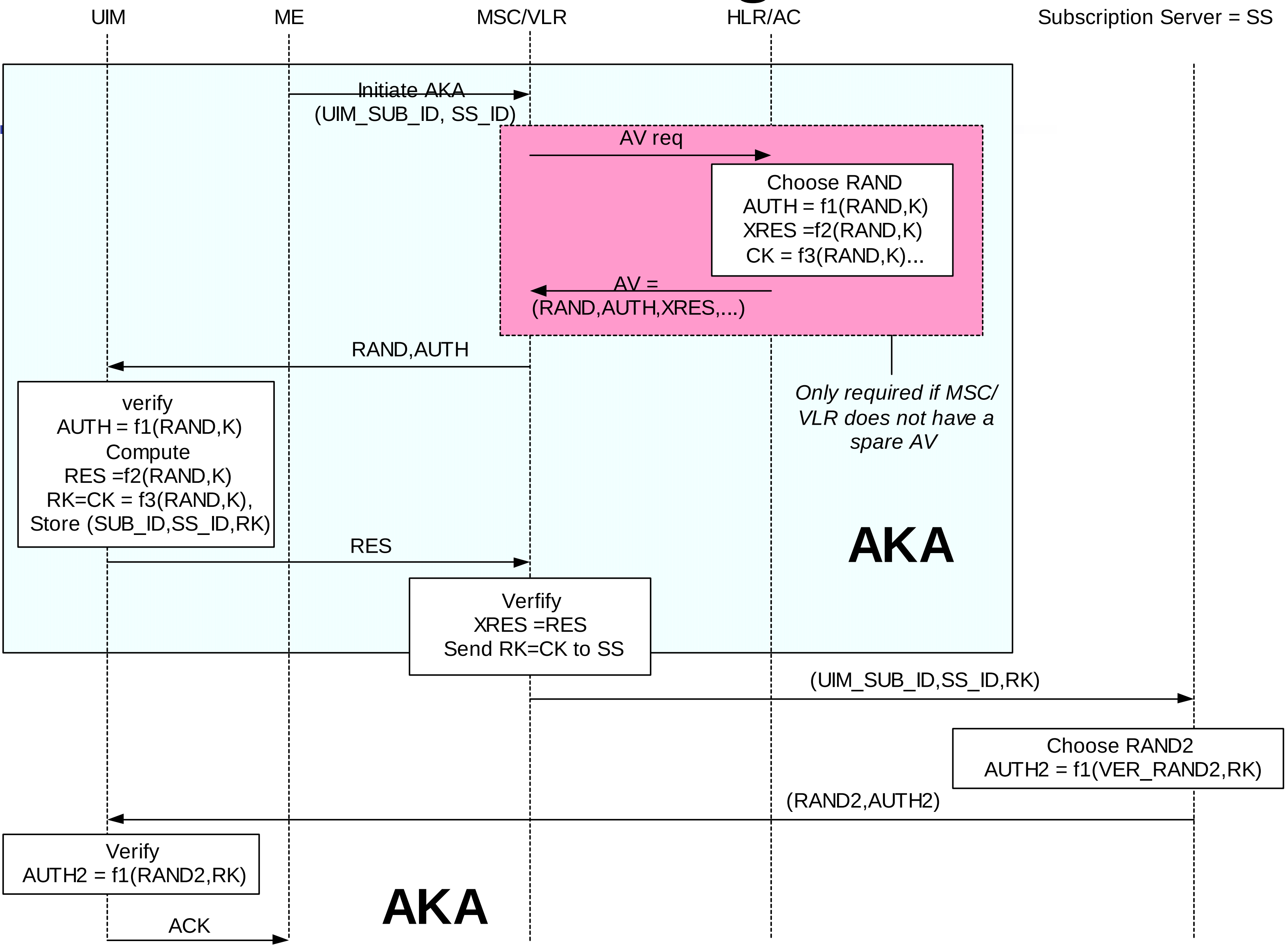
1. Content encryptor chooses SK, computes ESK and forms (BAKseq,ESK)
2. CE encrypts content with SK and sends (BAKseq,ESK) periodically with encrypted content.
3. ME passes (BAKseq,ESK) to UIM.
4. UIM computes SK passes SK to ME
5. ME decrypts with SK.

(BAKseq,ESK) sent in packet with unique port#. Maybe sync problems

Provisioning RK

- Previous submission
  - suggested using OTASP Secure Mode Cipher Key (SMCK) for encrypting BAK when sending to UIM
- Problem (?)…
  - How does BAK encryptor authenticate UIM?
  - No “security association” between SS and UIM
- If SS/UIM share RK, then key distribution process implicitly authenticates UIM
  - Use “special” AKA to provision RK

Provisioning RK



UIM

ME

MSC/VLR

HLR/AC

Subscription Server = SS

Initiate AKA
(UIM_SUB_ID, SS_ID)

AV req

Choose RAND
AUTH = f1(RAND,K)
XRES =f2(RAND,K)
CK = f3(RAND,K)...

AV =
(RAND,AUTH,XRES,...)

RAND,AUTH

Only required if MSC/ VLR does not have a spare AV

verify
AUTH = f1(RAND,K)
Compute
RES =f2(RAND,K)
RK=CK = f3(RAND,K),
Store (SUB_ID,SS_ID,RK)

RES

AKA

Verfify
XRES =RES
Send RK=CK to SS

(UIM_SUB_ID,SS_ID,RK)

Choose RAND2
AUTH2 = f1(VER_RAND2,RK)

(RAND2,AUTH2)

Verify
AUTH2 = f1(RAND2,RK)

ACK

AKA

Options

- BAK Update
- Encryption
- Location of BAK Encryptor
- SK Update

Options (1)

- BAK Update
  - 1. SS-encrypted. Steps:
    - BAK Encryptor sends BAK to SS
    - SS encrypts BAK and returns encrypted BAK to BAK encryptor.
  - 2. Locally encrypted
    - 2a.SS provides Temporary Key (TK) to local BAK encryptor
    - 2b. If BAK Encryptor in local network, then BAK encryptor may use TK=CK from authentication vector AV
- Encryption
  - 1. Link layer encryption
  - 2. End-to-end
    - 2a. IPSec (previous proposal)
    - 2b. Application layer

Options (2)

- Location of BAK Encryptor
    - 1. Associated with a single CS
    - 2. Centralized BAK encryptor (associated with many CSs)
- SK Update
    - SK derived from SPI
    - SK sent in encrypted form
        - Use special port number to indicate packet containing SK
        - Use special BSR_ID=000 to indicate packet containing SK

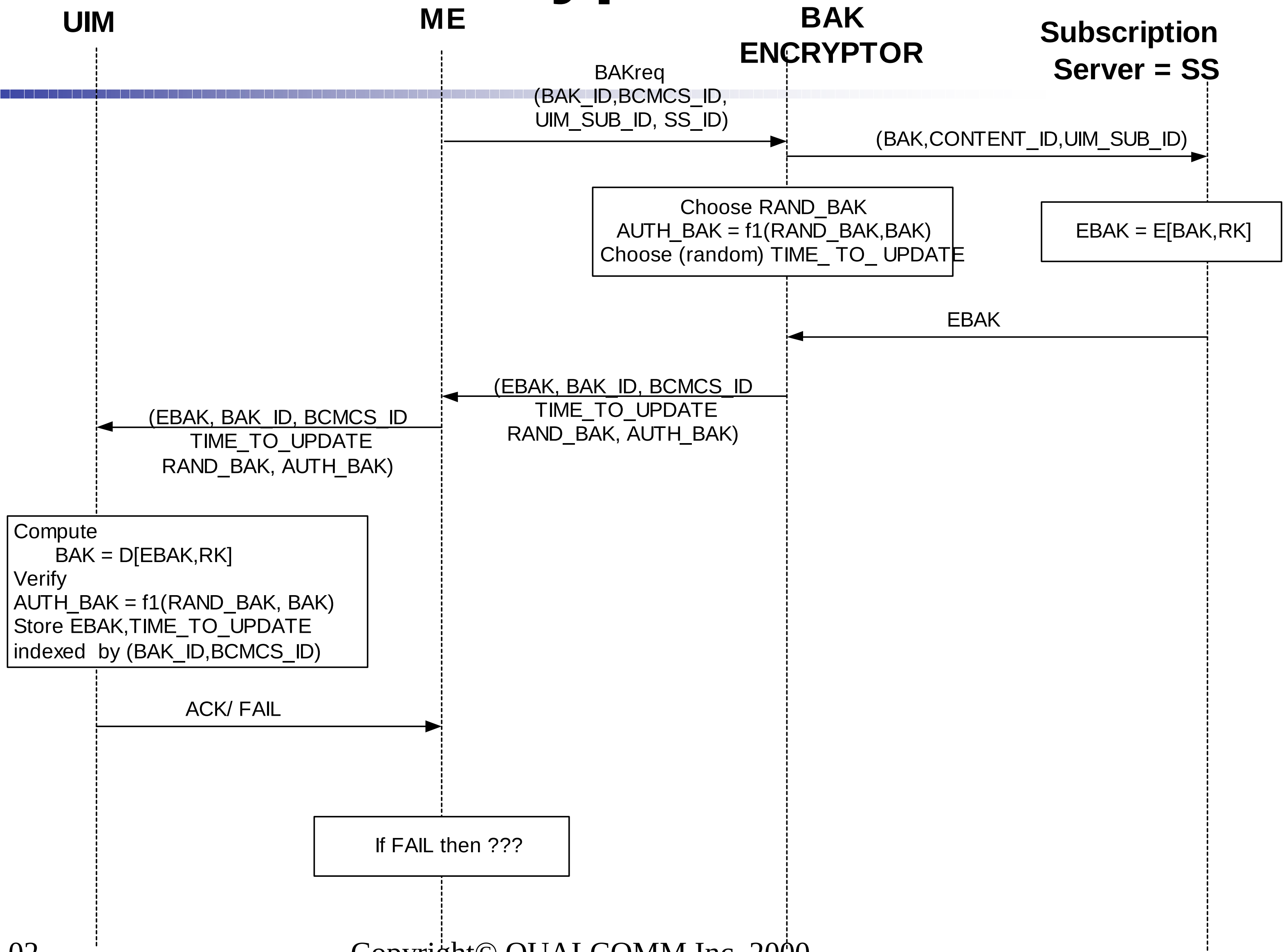
BAK Update

Options

1. SS Encrypted

- BAK encryptor sends BAK to SS
- SS encrypts BAK with RK to form EBAK and returns EBAK to BAK encryptor
- BAK encryptor sends EBAK to UIM
- UIM decrypts EBAK with RK to form BAK

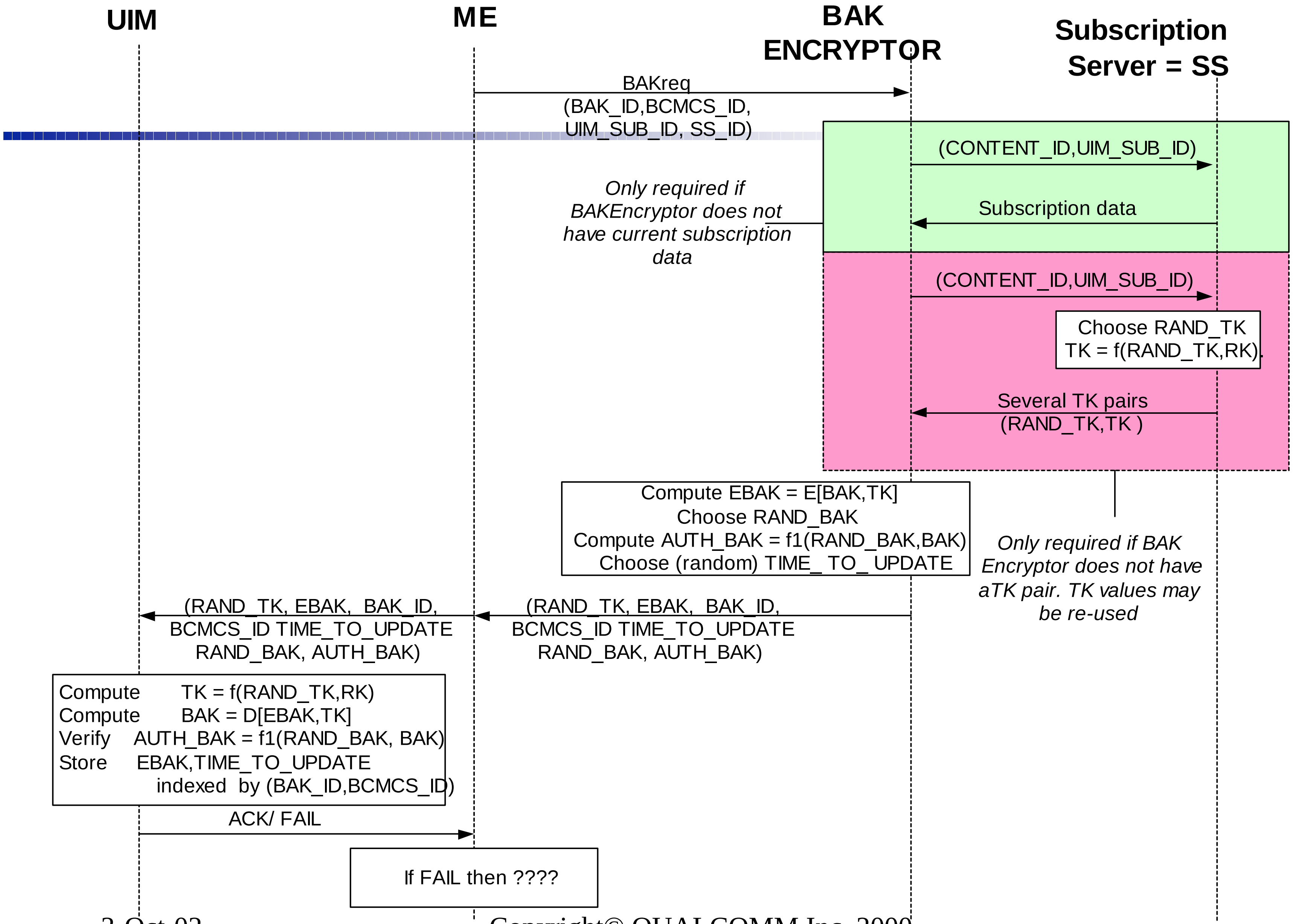
SS-Encrypted BAK


UIM
ME
BAK ENCRYPTOR
Subscription Server = SS
BAKreq (BAK_ID,BCMCS_ID, UIM_SUB_ID, SS_ID)
(BAK,CONTENT_ID,UIM_SUB_ID)
Choose RAND_BAK
AUTH_BAK = f1(RAND_BAK,BAK)
Choose (random) TIME_ TO_ UPDATE
EBAK = E[BAK,RK]
EBAK
(EBAK, BAK_ID, BCMCS_ID TIME_TO_UPDATE RAND_BAK, AUTH_BAK)
(EBAK, BAK_ID, BCMCS_ID TIME_TO_UPDATE RAND_BAK, AUTH_BAK)
Compute
BAK = D[EBAK,RK]
Verify
AUTH_BAK = f1(RAND_BAK, BAK)
Store EBAK,TIME_TO_UPDATE
indexed by (BAK_ID,BCMCS_ID)
ACK/ FAIL
If FAIL then ???

2a. BAK Locally Encrypted with Temporary Key (TK) from SS

- BAK encryptor obtains (TK_RAND,TK) pair from SS
  - TK = f(TK_RAND,RK)
  - BAK encryptor may re-use (TK_RAND,TK) pair, SS may send multiple pairs
- BAK encryptor
  - encrypts BAK with TK to form EBAK and
  - Sends (TK_RAND,EBAK) to UIM
- UIM forms TK = f(TK_RAND,RK)
- UIM decrypts EBAK with TK to form BAK

QUALCOMM



UIM

ME

BAK ENCRYPTOR

Subscription Server = SS

BAKreq (BAK_ID,BCMCS_ID, UIM_SUB_ID, SS_ID)

(CONTENT_ID,UIM_SUB_ID)

Subscription data

Only required if BAKEncryptor does not have current subscription data

(CONTENT_ID,UIM_SUB_ID)

Choose RAND_TK
TK = f(RAND_TK,RK).

Several TK pairs (RAND_TK,TK)

Compute EBAK = E[BAK,TK]
Choose RAND_BAK
Compute AUTH_BAK = f1(RAND_BAK,BAK)
Choose (random) TIME_ TO_ UPDATE

Only required if BAK Encryptor does not have aTK pair. TK values may be re-used

(RAND_TK, EBAK, BAK_ID, BCMCS_ID TIME_TO_UPDATE RAND_BAK, AUTH_BAK)

(RAND_TK, EBAK, BAK_ID, BCMCS_ID TIME_TO_UPDATE RAND_BAK, AUTH_BAK)

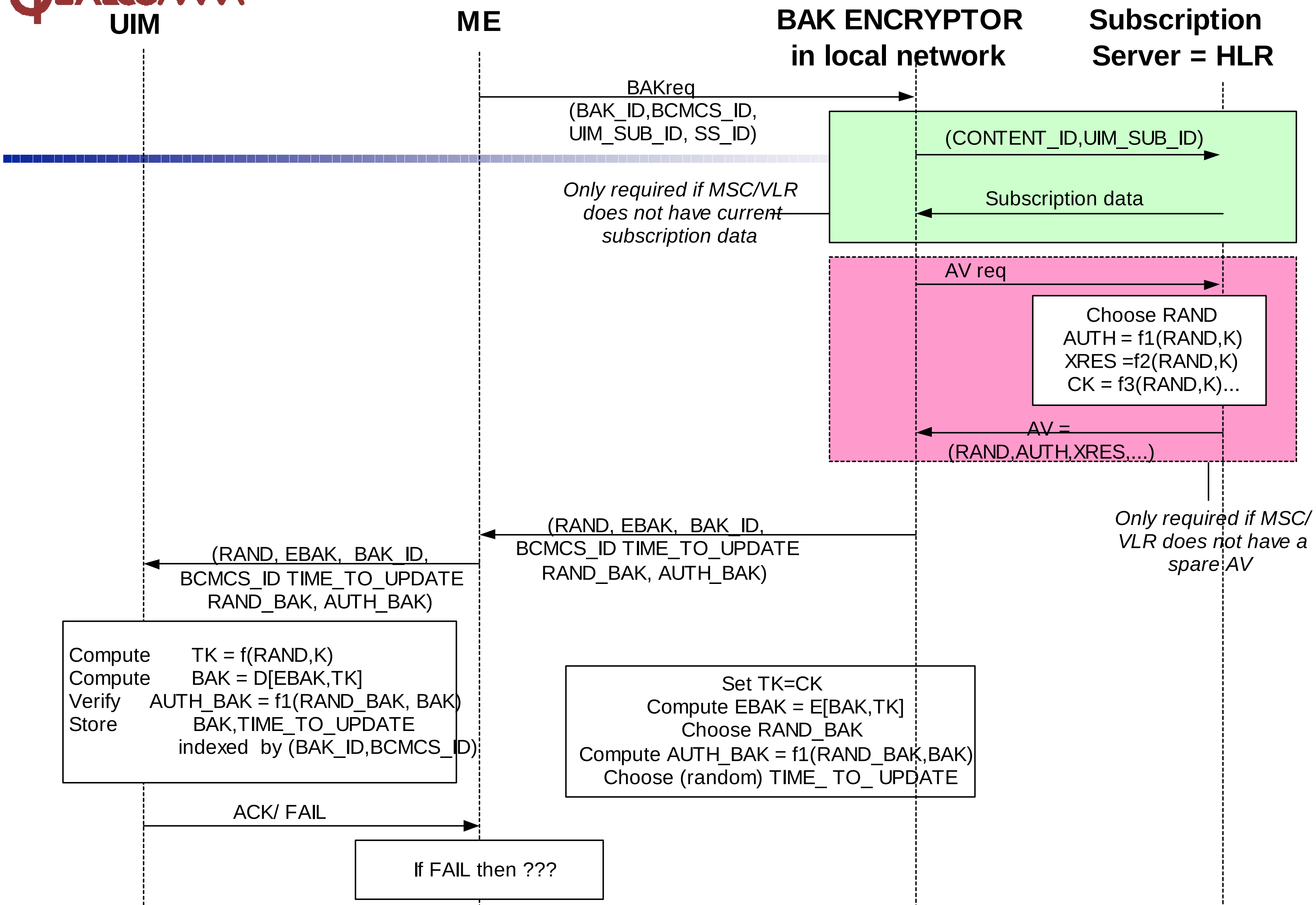
Compute TK = f(RAND_TK,RK)
Compute BAK = D[EBAK,TK]
Verify AUTH_BAK = f1(RAND_BAK, BAK)
Store EBAK,TIME_TO_UPDATE
indexed by (BAK_ID,BCMCS_ID)

ACK/ FAIL

If FAIL then ????

2b. BAK Locally Encrypted with key from AV

- Subscription Server = HLR/AAA
- MS performs special AKA with MSC/VLR
  - MSC/VLR can re-use AV or request AV from HLR/AC if not spare AV
  - AV includes CK = f(CK_RAND,K), K= A-key root key at HLR.
  - MSC/VLR sends (CK_RAND,CK) pair to BAK encryptor
- BAK encryptor
  - encrypts BAK with CK to form EBAK and
  - sends (CK_RAND, EBAK) to UIM.
- UIM forms CK = f(CK_RAND,K) as for AKA
- UIM decrypts EBAK with CK to form BAK
- Prevents "double requests" from HLR/AAA as uses available AV's.



UIM

ME

BAK ENCRYPTOR in local network

Subscription Server = HLR

BAKreq (BAK_ID,BCMCS_ID, UIM_SUB_ID, SS_ID)

(CONTENT_ID,UIM_SUB_ID)

Subscription data

Only required if MSC/VLR does not have current subscription data

AV req

Choose RAND
AUTH = f1(RAND,K)
XRES =f2(RAND,K)
CK = f3(RAND,K)...

AV = (RAND,AUTH,XRES,...)

Only required if MSC/ VLR does not have a spare AV

(RAND, EBAK, BAK_ID, BCMCS_ID TIME_TO_UPDATE RAND_BAK, AUTH_BAK)

(RAND, EBAK, BAK_ID, BCMCS_ID TIME_TO_UPDATE RAND_BAK, AUTH_BAK)

Compute TK = f(RAND,K)
Compute BAK = D[EBAK,TK]
Verify AUTH_BAK = f1(RAND_BAK, BAK)
Store BAK,TIME_TO_UPDATE indexed by (BAK_ID,BCMCS_ID)

Set TK=CK
Compute EBAK = E[BAK,TK]
Choose RAND_BAK
Compute AUTH_BAK = f1(RAND_BAK,BAK)
Choose (random) TIME_ TO_ UPDATE

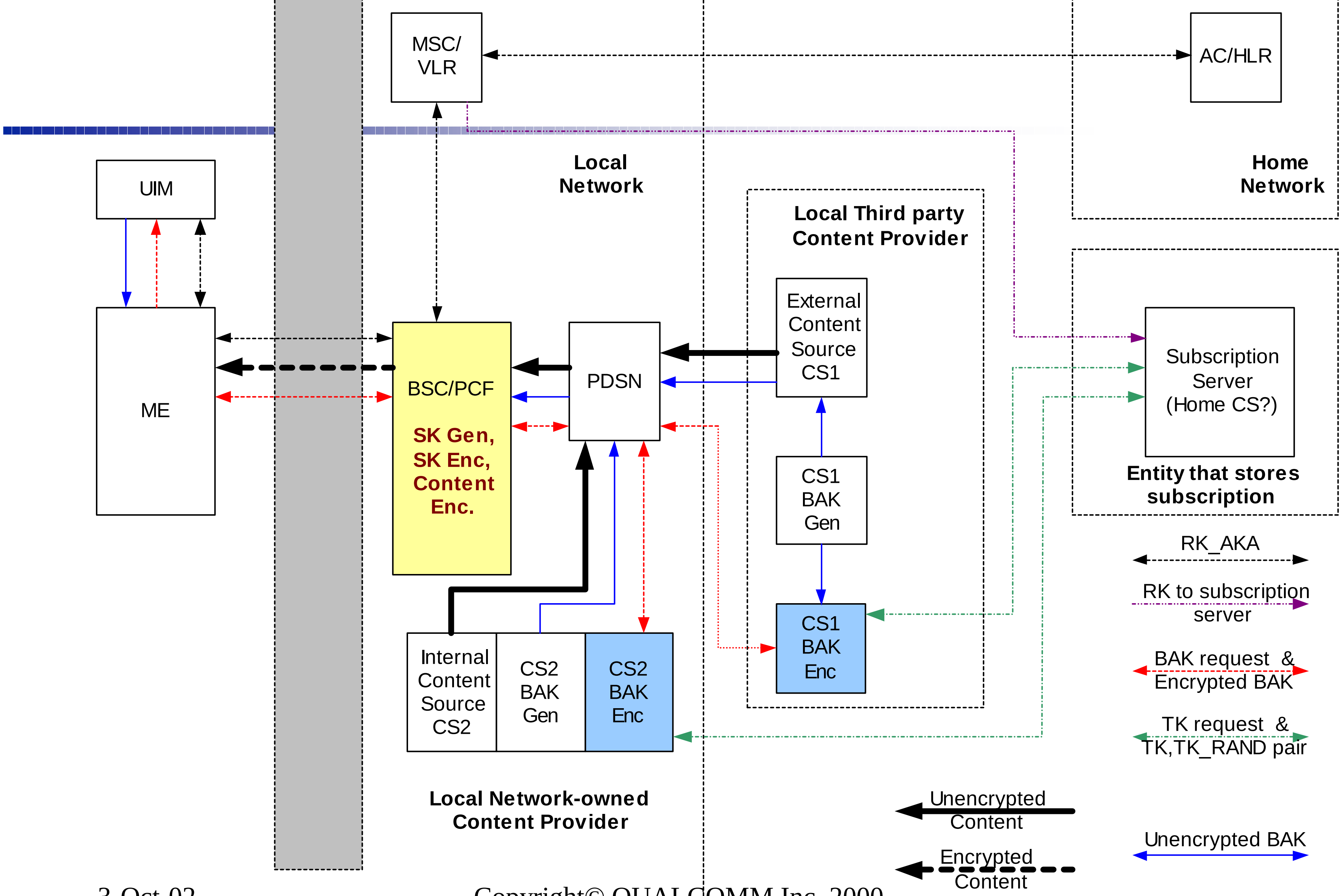
ACK/ FAIL

If FAIL then ???

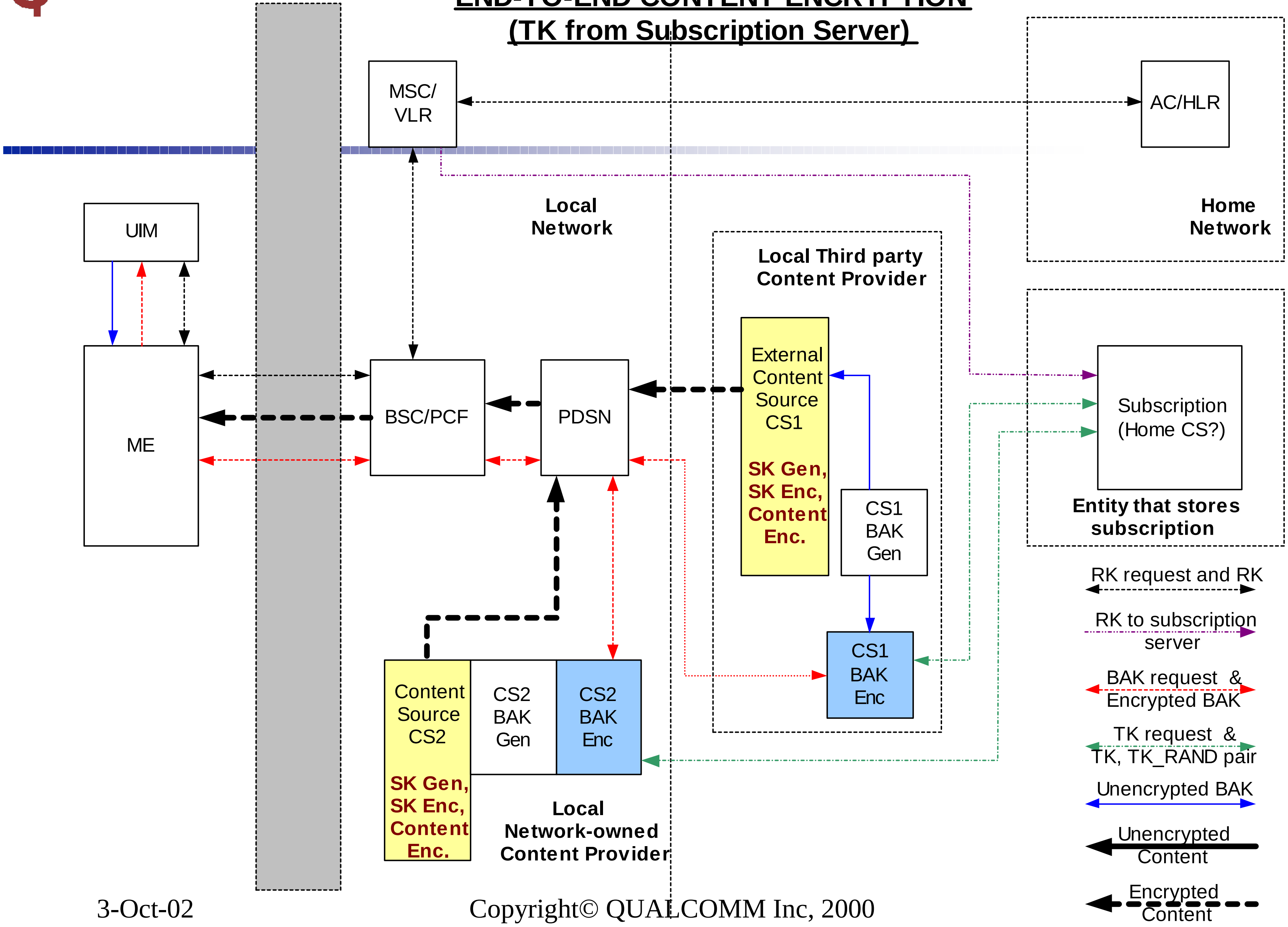
Encryption

Options

LINK LAYER CONTENT ENCRYPTION (TK from SS)


MSC/ VLR
AC/HLR
Local Network
Home Network
UIM
ME
BSC/PCF
SK Gen, SK Enc, Content Enc.
PDSN
Local Third party Content Provider
External Content Source CS1
CS1 BAK Gen
CS1 BAK Enc
Subscription Server (Home CS?)
Entity that stores subscription
Internal Content Source CS2
CS2 BAK Gen
CS2 BAK Enc
Local Network-owned Content Provider
RK_AKA
RK to subscription server
BAK request & Encrypted BAK
TK request & TK,TK_RAND pair
Unencrypted Content
Encrypted Content
Unencrypted BAK

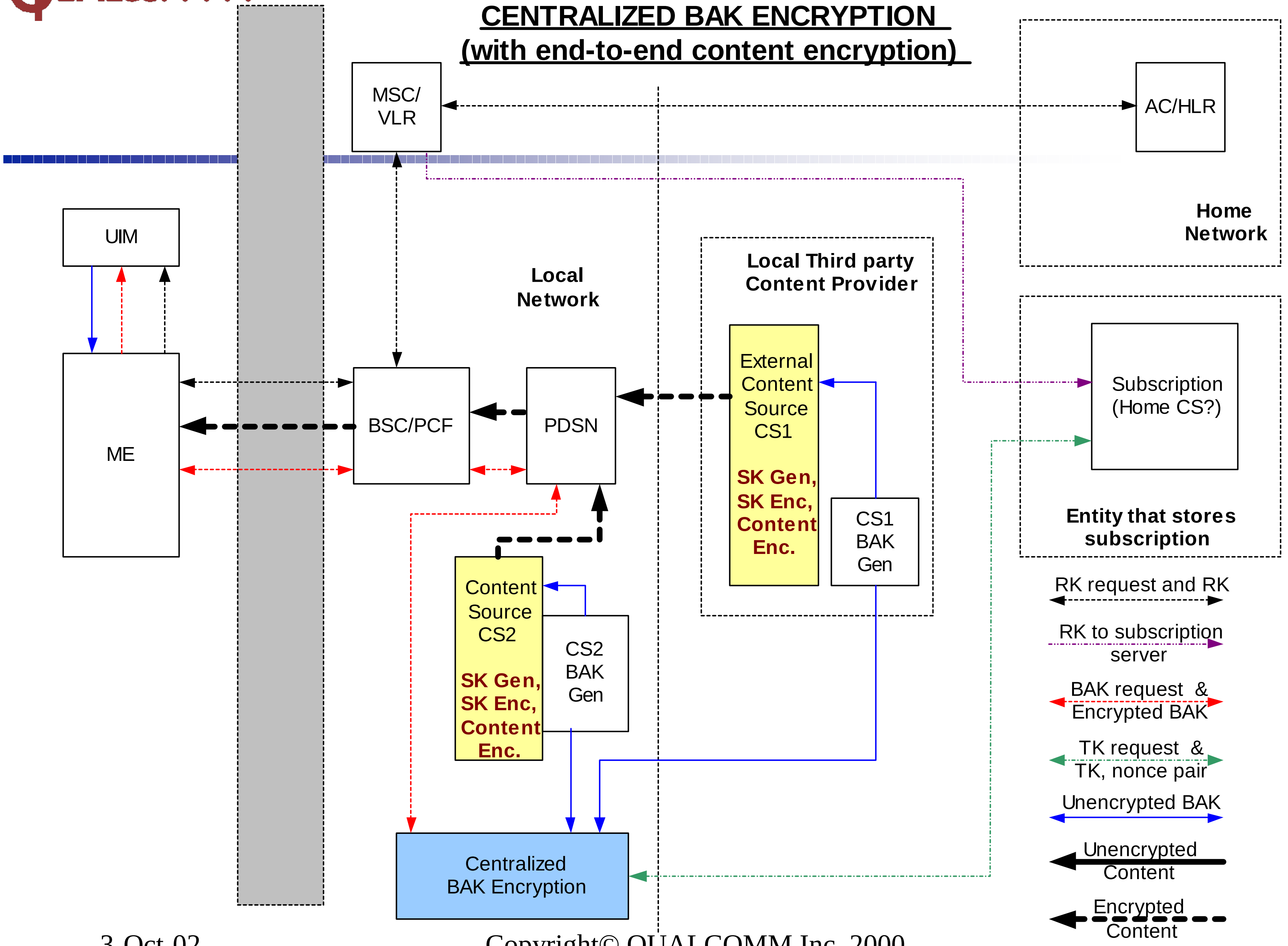
END-TO-END CONTENT ENCRYPTION (TK from Subscription Server)


MSC/ VLR
AC/HLR
UIM
ME
BSC/PCF
PDSN
Local Network
Home Network
Local Third party Content Provider
External Content Source CS1
SK Gen, SK Enc, Content Enc.
CS1 BAK Gen
CS1 BAK Enc
Subscription (Home CS?)
Entity that stores subscription
Content Source CS2
SK Gen, SK Enc, Content Enc.
CS2 BAK Gen
CS2 BAK Enc
Local Network-owned Content Provider
RK request and RK
RK to subscription server
BAK request & Encrypted BAK
TK request & TK, TK_RAND pair
Unencrypted BAK
Unencrypted Content
Encrypted Content

Location of BAK Encryptor

- Previous diagrams had a BAK Encryptor for each CS
- May also have centralized BAK Encryptor

CENTRALIZED BAK ENCRYPTION
(with end-to-end content encryption)



MSC/ VLR

AC/HLR

Home Network

UIM

ME

Local Network

BSC/PCF

PDSN

Local Third party Content Provider

External Content Source CS1

SK Gen, SK Enc, Content Enc.

CS1 BAK Gen

Subscription (Home CS?)

Entity that stores subscription

Content Source CS2

SK Gen, SK Enc, Content Enc.

CS2 BAK Gen

Centralized BAK Encryption

RK request and RK

RK to subscription server

BAK request & Encrypted BAK

TK request & TK, nonce pair

Unencrypted BAK

Unencrypted Content

Encrypted Content

SK Update

- SK derived from SPI
- SK sent in encrypted form

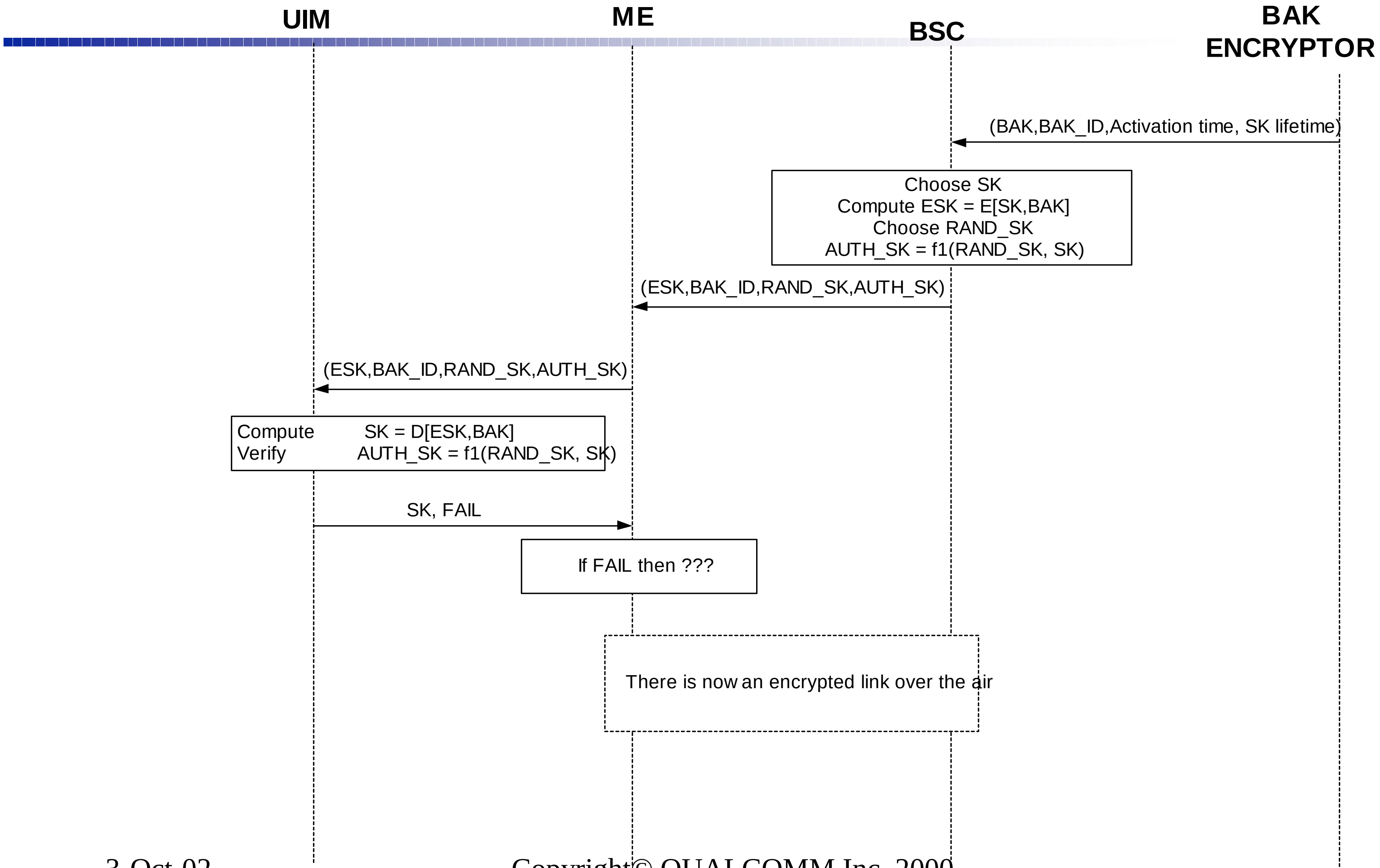
SK derived from SPI

- Proposed in previous
- SK generator
  - SPI in IPSec header
  - SK generator chooses SPI
    - 4-bits = BAK_ID / BAKseq
    - 28-bits = SPI_RAND
  - SK generator computes
    - SK = E[SPI_RAND,BAK]
  - SK gnerator uses SK to encrypt packets with this SPI in header
- UIM
  - re-generates SK from SPI
  - uses SK to decrypt

SK sent in encrypted form

- SK generator
  - chooses SK
  - Computes ESK = E[SK,BAK]
  - Sends ESK as separate packet
- Differentiating packets containing SK
  - special port number
  - Use special BSR_ID=000
- Problems
  - Synchronization
    - (not a problem for link-layer encryption)

SK-update for Link-layer encryption



UIM
ME
BSC
BAK ENCRYPTOR
(BAK,BAK_ID,Activation time, SK lifetime)
(ESK,BAK_ID,RAND_SK,AUTH_SK)
(ESK,BAK_ID,RAND_SK,AUTH_SK)
SK, FAIL

Choose SK
Compute ESK = E[SK,BAK]
Choose RAND_SK
AUTH_SK = f1(RAND_SK, SK)

Compute SK = D[ESK,BAK]
Verify AUTH_SK = f1(RAND_SK, SK)

If FAIL then ???

There is now an encrypted link over the air