3GPP TSG SA WG3 Security — SA3#33 **S3-040258**
10-14 May 2004
Beijing, China

Source: **Ericsson**

Title: **Extension payloads to MIKEY to support MBMS**

Document for: **Discussion**

Agenda Item: **MBMS**

1 Introduction

This is an update of S3-040235 due to the comments that MIKEY solution has not specified the UICC solution.

The MIKEY enhancements presented in S3-040081 [2] introduced unnecessary deviation from original MIKEY [1]. To minimize the impact on the MIKEY protocol, this document describes how the same information can be conveyed to the ME by means of the extension payload present in MIKEY. The purpose of this payload is, as the name suggests, to enable extensions to the protocol in a controlled fashion.

2 Usage of MIKEY header

MIKEY can be used to transport the MSK as well as the MTK to the UE.

The MIKEY common header payload is used as specified in [1] with the exception of additional data types used in the Data Type Field.

The **CSB ID** defines the specific MBMS service.

The ME can detect from the **Data Type** field of the MIKEY header whether the message transports a MSK or MTK. Two values are (given in hexadecimal):

Data Type	Meaning
0x07	The key in the MIKEY message is an MSK and is encrypted with a MUK
0x08	The key in the MIKEY message is an MTK and is encrypted with an MSK

2 The General Extension Payload

The general extension payload of MIKEY was defined to enable extensions to MIKEY in a controlled way, so that possible error conditions arising from non-supported extensions can be handled in a graceful way. The payload format has the following form:

Next Payload	**Type**	**Length**
Data		

where **Next Payload** is an 8-bit field identifying the payload after this one, **Type** is an 8-bit field specifying the type of data present in the **Data** field, and **Length** is a 16-bit field specifying the number of octets of the **Data** field.

The two values 0 and 1 are defined for the **Type** field in [1].

3 Usage of the General Extension Payload

The MIKEY in [2] introduced unnecessary deviation from original MIKEY [1] and can cause difficulties for implementations unaware of the extensions used in MBMS. The same information can be conveyed from the BM-SC to the ME by the use of the general extension payload. This approach has the benefit that legacy implementations can parse the messages unambiguously and discard unsupported ones.

Let the fields **Next Payload** and **Length** be as defined in [1] and the **Type** field be 2 (this should however be registered with IANA if time permits). The information needed for the combined method of key management in MBMS can now be carried in the **Data** field. The structure of the **Data** field is as follows:

Outer Key ID	**Inner Key ID**

where **Outer Key ID** (16 bits) is the identity of the key used to encrypt the Key Data Sub Field in KEMAC and **Inner Key ID** is the identity of the key in the Key Data field in the Key Data Sub Payload. In the case there are more than one key in the KEMAC payload, there must be equally many ID:s in the extension payload. Furthermore, the order in which the keys are listed must correspond to the order of the list in the extension payload. In case there is not a one-to-one mapping between the two lists, the entire MIKEY message should be discarded.

The ME could use the Key ID:s to make sure that it does not try to install a key that is already in place. It is possible that the same key is delivered multiple times over the broadcast channel for reliability reasons. This would be seen as a replay attack (see Section 5), and it is also inefficient to decrypt and authenticate a message with already known information. This of course puts the requirement on the ME to keep track of which keys are installed.

4 Usage of the KEMAC Payload

The KEMAC payload is used as specified in [1] with the exception of additional key types used in the Key Data Sub Field.

To distinguish the ``level´´ of the keys, the **Type** field is used. It has one of the following values (given in hexadecimal):

Key Type value	Meaning
0x04	The key in the **Key** field is an MSK and is encrypted with a MUK
0x05	The key in the **Key** field is an MTK and is encrypted with an MSK
0x06	The key in the **Key** field is an MTK for encryption and is encrypted with an MSK
0x07	The key in the **Key** field is an MTK for integrity protection and is encrypted with an MSK

The value 0x05 is to be used if key derivation (splitting of a key into one encryption and one integrity key) is to be done in the ME. Note that the idea of splitting keys is to create two or more cryptographically independent keys, where the compromise of one of them will not reveal any information about any of the other. The new type values should be registered with IANA if time allows it.

When the UE receives a MIKEY message containing an MSK, it uses a MUK_A (derived from MUK) to authenticate the message, and a MUK_E (derived from MUK) to decrypt the KEMAC payload.

When the UE receives a MIKEY message containing an MTK, it uses a MSK_A (derived from MSK) to authenticate the message, and a MSK_E (derived from MSK) to decrypt the KEMAC payload.

The optional **Key Validity Data** subfield is present in the KEMAC payload when MSK is transported but it is not present for MTK transport. The field defines the Key Validity Time for MSK in terms of sequence numbers (i.e. MTK IDs). The lower limit of the interval defines the SEQs in the MTK Generation and Validation Function in TS 33.246.

An example of a (logical) MIKEY message that transports an MSK to the UE is shown in Figure 2. The MUK and MSK ID:s are left unencrypted, so that the ME can check if the MSK is already installed or not.

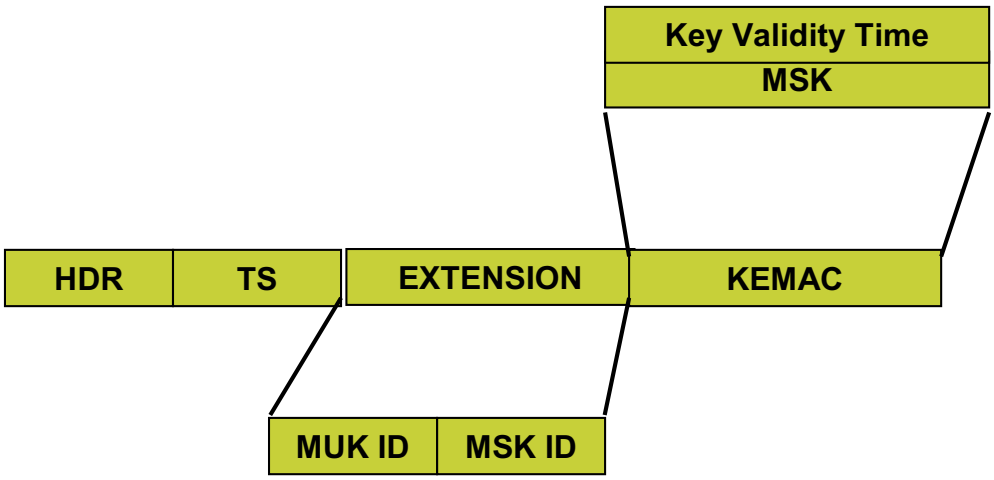

Figure 1. Example showing a (logical) MIKEY message transporting a MSK to the UE.

5 Replay Protection

The MIKEY message is protected for replay using the standard MIKEY mechanism by specifying the time stamp payload in [1] to be of type 2 (COUNTER). This mechanism can be used in the UICC based model as well as in the ME based model. The replay counter will be different to MSK and MTK transport.

6 Acknowledgement Messages

In some cases the server may want an acknowledgement message. We will use the commonly used terminology ACK. Since we propose to use MIKEY, it is natural to also use the MIKEY response mechanism also for the replies. This means that the ME would return a message looking like below (on a high level):

Figure 2. Example showing a (logical) MIKEY acknowledgement message.

where V is the verification payload containing a MAC covering the HDR and TS fields. The TS field would contain the same TS (the counter) as received from the BM-SC.

7 UICC model Issues

In the UICC model, the MSK is kept secret from the ME inside the UICC. When this is the case, the integrity of the MIKEY message is verified in the UICC. This implies that the entire MIKEY

message is passed to the UICC since the MAC in the KEMAC payload is used. It should be noted that this is done for both MSK and MTK level MIKEY messages.

The interface between the UICC and the ME needs a way to return the success/failure of the MAC verification to the ME. This also enables the ME to send ACK messages back to the BM-SC if required. In the case that the ME needs to send a response to the BM-SC, it is also required that the interface to the UICC is capable of computing the MAC of the response MIKEY message. Typically, the ME would create the message, filling in all fields except for the MAC field. After this the ME would send the message to the UICC for MAC computation, and the UICC would return the MAC to the ME for insertion in the message.

8 Example functional walkthrough

This chapter gives a high level view how MIKEY messages carrying MSK or MTK keys are handled in the UE.

8.1 MSK transport

It is assumed that the UE includes the secure storage (MGV-S) that is defined in chapter 6.3 of TS 33.246. The MGV-S may be realized on the ME or on the UICC. It is also assumed that the MGV-S includes the MUK keys MUK_A and MUK_E. How these are derived is described in xxx.

The following is an example walkthrough of MSK transport.

1. When the ME receives MIKEY message it performs a sanity check for the header fields and checks the replay protection of the TS payload. It may also check the Key IDs in the extension payload so that it does not try to install a key that is already in place. If some of these checks fail, the message is discarded.
2. The ME sends the message to the MGV-S for integrity checking.
3. The MGV-S performs the integrity check of the whole message using MAC in the KEMAC payload and MUK_A. If the integrity check fails, processing is aborted and MGV-S indicates a failure to the ME.
4. The MGV-S decrypts the encrypted content of KEMAC payload with MUK_E and stores the key information, e.g. Service ID, MSK ID, MSK and MSK Validity Time. The lower limit of Key Validity Time is the SEQs defined in section 6.3 of TS 33.246.
5. The MGV-S sends the integrity check result to the ME.
6. If the integrity check was successful, the ME stores the key information, e.g. Service ID, MUK ID, MSK ID. If the MGV-S indicated failure, the message is discarded.

8.2 MTK transport

The following is an example walkthrough of MTK transport.

1. When the ME receives MIKEY message it performs a sanity check for the header fields and checks the replay protection of the TS payload. It may also check the Key IDs in the extension payload so that it does not try to install a key that is already in place. If some of these checks fail, the message is discarded.
2. The ME sends the message to the MGV-S for validation and MTK generation.
3. The MGV-S performs the MGV-F (MTK Generation and Validation Function) for the MIKEY message (see Note 1). If the validation fails, processing is aborted and MGV-S

indicates a failure to the ME. If the validation is successful, the MGV-S indicates success to the ME and delivers the corresponding MTK to the ME.

4. If the validation was successful, the ME stores the MTK information, e.g. MTK and MTK-ID. If the MGV-S indicated failure, the MIKEY message is discarded.

Note 1: The MGV-F in section 6.3 of TS 33.246 needs to be changed. Instead of having encrypted MTK and SEQ as the input to the validation calculation, the new input is the whole MIKEY message, which among other fields includes the encrypted MTK and the MTK ID as the SEQ. The result of the validation is compared to the MAC of the KEMAC payload. For the updated MGV-F see pseudo CR xxx.

9 Conclusions

The document has described how MBMS key management can be achieved with MIKEY in both ME and UICC based models. It is proposed that MIKEY is chosen as the key management method for MBMS.

10 References

[1] Arkko et. al., ``draft-ietf-msec-mikey-08.txt´´, IETF draft, 2003, work in progress
[2] Nokia, ``S3-040081: Use of MIKEY in Combined method´´