3GPP TSG SA WG3 Security — S3#34 **S3-040541**
6 - 9 July 2004
Acapulco, Mexico

Source: **Ericsson**

Title: **Harmonized Key Management for Streaming and Download**

Document for: **Discussion and decision**

Agenda Item: **MBMS**

1 Introduction

This paper discusses how the key management in MBMS can be performed using MIKEY [2] in more detail, and shows that the same mechanism can be used to deliver keys both for download and for streaming.

2 Harmonized Streaming and Download Key Management

This section will describe how the same key management (and level of keys) can be used to provide keys for both download and streaming.

2.1 Key Distribution

The establishment of keys can be described as follows. The key management sequence starts with the terminal establishing a key (from which the MUK can be derived) with the BSF. Once the key is in place, the terminal contacts the NAF (the BM-SC in our case). The BM-SC needs to associate an ID with the components that are to use the same MSK. There may be cases where one or several components (e.g., a video- and an audio stream) can use the same MSK. The set of components that share the same MSK will be referred to as a component set further on.

At this point the terminal and BM-SC can start key management activities for MSK:s and MTK:s (which are carried in MIKEY as specified in detail in Section 3).

The MSK:s are protected by the MUK:s, and the MTK:s are protected by the MSK:s. Note that there is a need for separate encryption, authentication and salting keys. The salting keys are required for any stream cipher [1] under certain circumstances, e.g., AES in Counter mode (note that AES is a block cipher, but when run in Counter mode it effectively becomes a stream cipher). These three types of keys are either delivered separately or as one key that is later driven through a key derivation function.

2.2 Key Reception

Once the keys are decrypted and verified, they are installed in the key database (indexed by component set ID and key ID). The key ID:s should be represented by unsigned integer counters. Again, see Section 3 for the details. MIKEY would typically install a single MSK and MTK, and

when the keys are retrieved from the key database, they are split into several keys if required by the security protocol. The key ID:s are only required to be unique per key type (MSK or MTK), since they are local to each component set ID. Note that the MIKEY implementation is agnostic of the socket on which the message arrived; this implies that the MSK:s can be delivered over an MBMS bearer or any other bearer.

2.3 Key Usage by Security Protocols

As far as the security protocols are concerned, they do not have to be aware of the MSK:s. They simply ask the MIKEY key database for the key with the given ID (which is the concatenation of the MSK ID and the MTK ID and possibly something more.

When a usable MTK is installed in the key database it can be passed to the streaming protection mechanism and the un-protection can proceed as expected. In the download case, the MTK is passed as the pre-shared key to the S/MIME implementation. It is an implementation issue how this is actually achieved, since both the S/MIME and streaming protection mechanism can have access to the key database where the keys were installed. They would then both retrieve the ID of the key to use from the received data (from the S/MIME container in the download case (see [3]), and from the MKI if SRTP is chosen for streaming).

There might be a different set of parameters associated with each key (a security association (SA)). The SA has different content depending on which protocol it is intended for. This means that, e.g., an MSK intended for download would have a certain type of SA, which cannot be used in the streaming case. This is however not a problem for the key management, that only deals with the delivery of the SA:s and keys, and not with the intended usage of the keys. In a slightly obscure scenario where a particular key is to be used with different security protocols, the same key might have to be delivered twice, to allow the different parameters to be installed in the terminal.

Note that there is no difference in the key distribution process for download and streaming.

The layer that initiates the security protocol can perform key derivation on the single key if the security protocol used requires separate encryption, salting or authentication keys.

3 Details of MIKEY Messages Construction and Parsing

In this section it is assumed that the MUK (which is put in place at the terminal and at the BM-SC by use of GBA or other mechanism) is put through a key derivation function (e.g., the one specified in MIKEY [2] Section 4.1.4) so that there is now one MUK integrity key MUK_I, one MUK confidentiality key MUK_C and one MUK salting key MUK_S (this corresponds to applying the functions F_f and F_g to the MUK, i.e., MUK_I = MFK, MUK_C = MGK and MUK_S is not in TS33.246 yet but is needed if stream-cipher encryption is to be used, e.g., AES-CM [1]). Note that all steps are not included in the following descriptions.

3.1 MSK MIKEY Message Construction

The BM-SC generates the key delivery message using MIKEY [2] by performing the following steps (note that the bullets in step 1 related to the CS ID Map are not relevant to MTK messages):

1. First the Common Header Payload (HDR) is created (Section 6.1 of [2]) with the following field-values:
   - Version = 1
   - Data Type = 7 (for MSK delivery, to be registered with IANA)
   - V = 1 if a response is required and 0 otherwise
   - PRF = 0 (the default MIKEY PRF)
   - CSB ID = randomly chosen (must be unique for each MBMS component set)
   - #CS = 1 (number of crypto sessions covered by this MSK)
   - CS ID Map Type = 0 (This means that there is an SRTP policy map sent. We need to specify another map and policy payload for the download protocol)
   - ***The rest of the bullets are only used for MSK delivery. Note that for MSK delivery the #CS field is 0, meaning that there will not be a CS ID Map Info payload present.***CS ID Map Info as specified below (note that this is only relevant for MSK:s and in the streaming case and if SRTP is used, in which case also a Security Policy Payload must be added). When #CS = 0, the CS ID Map Info will be the empty string (which is the case for MSK delivery).

Version	Data Type	Next Payload	V	PRF
CSB ID				
#CS	CS ID Map Type	CS ID Map Info		

Common Header Payload

The CS ID Map Info is made up of the following values:

- Policy No = 0
- SSRC = randomly chosen (Assumes only the streaming server is using the SSRC, otherwise uniqueness must be assured)
- ROC = the current value of the STP rollover-counter used by the BM-SC

Policy No	SSRC…
SSRC cont…	ROC…
ROC cont…	

CS ID Map Info

2. Next the Timestamp Payload (T) is created (Section 6.6 of [2]) using these values:
   - TS Type = 2 (meaning we will use a counter)
   - Counter = the value of an ``ever-increasing'' 32-bit counter that is bumped by one for each MIKEY message sent from the BM-SC to the terminal. Note that the counter wraps modulo 2^{32}.

Next Payload	TS Type	Counter

Timestamp Payload

3. Then the RAND Payload is created (Section 6.11 of [2]). Note that the RAND is only sent with the first MSK. It should be stored by the UE, and be reused for subsequently delivered MSK:s and MTK:s.
   - RAND Len = 16 (number of bytes in RAND-field)
   - RAND = (Pseudo) random string of data

Next Payload	RAND Len	RAND

RAND Payload

4. A Key data sub-payload is constructed as specified in Section 6.13 of [2], yielding the following payload:
   - Type = 2 (meaning the key is an TEK)
   - KV = 1 (meaning the key is associated with the ID present in the KV Data field).
   - Key Data Length = 16 (number of bytes in MSK)
   - VF Length
   - VT Length
   - Valid from, specified as MTK ID
   - Valid to, specified as MTK ID

Next Payload	Type	KV	Key Data Length
MSK Data			
VF length	Valid From (as MTK-ID)		
VT length	Valid To (as MTK-ID)		

Key Data sub-payload

5. Next, the Extension Payload (EXT) specified in S3-040258 is constructed.
6. Finally, a KEMAC Payload (Section 6.2 of [2]) is constructed. The encryption is performed using MUK_C and MUK_S (Section 4.2.3 of [2]), and the MAC is computed using MUK_I (Section 4.2.4 of [2]).
   - Encr Algo = 1 (meaning AES in Counter mode)
   - Encr Data Length = length of the Key Data sub-payload of bullet 4.
   - MAC Algo = 1 (meaning HMAC-SHA-1-160)
   - MAC = the MAC covering the entire MIKEY message (including the Extension payload)

Next Payload	Encr Algo	Encr Data Length
Encr Data		
MAC Algo	MAC	

KEMAC Payload

7. When all payloads are created, the Next Payload fields are filled in and message is put together as follows and is sent: Message = HDR || T || RAND || EXT || KEMAC.

If there would be separate delivery of MSK_I, MSK_C and MSK_S, bullet 4 would have to be repeated three times to create one Key Data sub-payload for each one of them. The key payloads would be placed in the order MSK_I, MSK_C and MSK_S.

3.2 MSK MIKEY Message Reception

When the MIKEY message arrives at the terminal, the processing proceeds following the steps below (basically following Section 5.3 of [2]).

1. The Data Type field of HDR is examined, and if it indicates an MSK, the MUK ID is extracted from the Extension Payload.
2. Check the Timestamp Payload, and discard the message if the Counter is larger or equal to the current MIKEY replay counter associated with the given MUK. To avoid issues with wrap around of the ID fields ``smaller than´´ can be in sense of RFC1982 [4].
3. Next MAC is verified using MUK_I, and the message is discarded upon failure (this corresponds to the F_m function).
4. If the MAC verification is successful the MUK_C and MUK_S is used to decrypt the Key Data sub-payload, and the MSK can be installed in the key database (this corresponds to the F_t function, except for the MUK_S which must also be used). The MSK can now be run through a key derivation mechanism (e.g., the one in Section 4.1.4 of [2]) to produce MSK_I, MSK_C and MSK_S unless all these three keys were delivered in separate Key Data sub-payloads.

3.3 MTK MIKEY Message Construction and Reception

The construction and processing of MUK messages are identical to the processing of MSK messages, with the only difference being that MUK is replaced by MSK and MSK is replaced by MTK. Furthermore, the Data Type in step 1 should be 8 to indicate MTK delivery, and the #CS should be 1. Note that by setting the #CS to 1, this enforces the CS ID Map Info sub-payload, which contains SRTP parameters which are required if SRTP is going to be used, and can be ignored otherwise (e.g., if the MTK is to be used for S/MIME).

4 IETF Considerations

It is clear that the usage of MIKEY in MBMS requires modifications to the protocol. With respect to the hard time constraints on MBMS and the time it takes to get work done in IETF, it is probably best to specify the changes in 3GPP and then submit drafts to IETF aiming for informal RFC:s. The hope is that the numbers used will be assigned by IANA.

In particular the following is important:

- UDP Port for MIKEY
- Data Types 0x07 and 0x08 for the MIKEY common header
- Security Policy payload formats for the security protocols used.

5 Conclusion and Proposal

This paper describes how MIKEY can be used in a key management mechanism which is common to both download and streaming. Ericsson also provides a pseudo CR that implements these changes.

6 References

[1] D. McGrew and S. Fluhrer, ``Attacks on Additive Encryption of Redundant Plaintext and Implications on Internet Security'', SAC 2000
[2] J. Arkko et. al., MIKEY, draft-ietf-msec-mikey-08.txt, IETF draft, work in progress
[3] S3-040xxx, MBMS Download Protection, Ericsson, SA3#34
[4] R. Elz and R. Bush, Serial Number Arithmetic, IETF RFC1982