3GPP TSG SA WG3 Security — S3#25 **S3-020535**
8 - 11 October 2002
Munich, Germany

Source: Ericsson

Title: MBMS - re-keying

Document for: Discussion and approval

Date: 2-10-02

Scope

This discussion paper focuses on re-keying of group members in an MBMS session. It includes some arguments and scenarios that motivate the use of re-keying. Further, it presents common approaches to re-keying and how one of them can be integrated in the key-management protocol MIKEY.

Abbreviations

These abbreviations are used throughout the document. The list is provided for readers not familiar with security terminology. For definitions that are more exact, we refer mainly to (www.ietf.org).

- *IETF* Internet Engineering Task Force, a standardization body
- *LKH* Logical Key Hierarchy
- *DRM* Digital Rights Management
- *MBMS* Multicast Broadcast Multimedia Services (3GPP service
- *KEK* Key Encrypting Key
- *TEK* Traffic Encrypting Key
- *MAC* Message Authentication Code

Introduction

MBMS sessions typically consist of a set of users receiving multicast/broadcast data. It is expected that the users will have to pay some fee to participate in an MBMS session. To protect the multicast data from others than the paying users, it is encrypted under some key (TEK). When a user decides to leave the group the rest of the users have to be supplied with a new TEK, i.e. they need to be re-keyed. The re-keying must of course be done in an efficient manner.

It can be argued that re-keying is not strictly necessary. That is, in some business models it would suffice to perform the keying that is achieved during registration. But re-keying opens up for other models. For instance, assume that a session is continuously running 24 hours a day, e.g. a certain music channel streaming music videos. Then it would make sense if users pay for the time they are members of the session. That is, a user joins the MTV session by running a registration protocol, e.g. MIKEY[1], and retrieves the current TEK. After 20 minutes the user decides to depart. At this point the same TEK can no longer be used, since the user still has it, and hence could decrypt the data belonging to the session. Hence, the TEK must be replaced by a new one, i.e. the remaining users need to perform re-keying.

Re-keying might also add inconvenience for malicious users. Assume that a user subscribes to the MBMS service, but once the user gets the TEK he publishes the key on a web-site. In this case, regular re-keying will at least make it more difficult for the malicious user. Also, there is probably a need for more elaborate DRM mechanisms to get content providers interested in MBMS; this should be further studied.

Even though the above scenario may not be implemented in the first phase of MBMS existence. It is beneficial to have the functionality present in the mobile terminals from day one, since upgrading them at a later time might be cumbersome.

Re-keying Methods

There are several ways to perform re-keying in group scenarios. In this section they are divided in two categories, unicast and multicast re-keying. The basic functionality is briefly described for each case.

Unicast Based Re-keying

The naive approach to re-keying is that the MB-SC exchanges keys with each member of the group individually. That is, when it is time for re-keying, the new TEK could be pushed down to each host, protected by their uniquely shared KEK. This solution requires that the BM-SC keeps a separate KEK for each member of the MBMS session. Furthermore, to exclude a set of members the MB-SC needs to perform unicast interaction with each of the remaining members.

In the unicast case, the authenticity of the re-keying message could either be deduced from that only the MB-SC could have encrypted the message under the given KEK so that the decrypted message makes sense. If a stronger authentication check is desired, a MAC based on some shared secret could be appended to the message.

Multicast Based Re-keying

Another approach to re-keying, is to use multicast messages to send the new TEK to several members of the group simultaneously. A common way to design multicast based re-keying is to use mechanisms like the Logical Key Hierarchies (LKH) [3] or similar.

Figure 1 depicts the structure of a small set of users and the LKH that can be used to re-key them. In the figure, the group controller (MB-SC) is located at the root of the tree and the group members show up as the leaves of the tree (numbered 1 to 8).

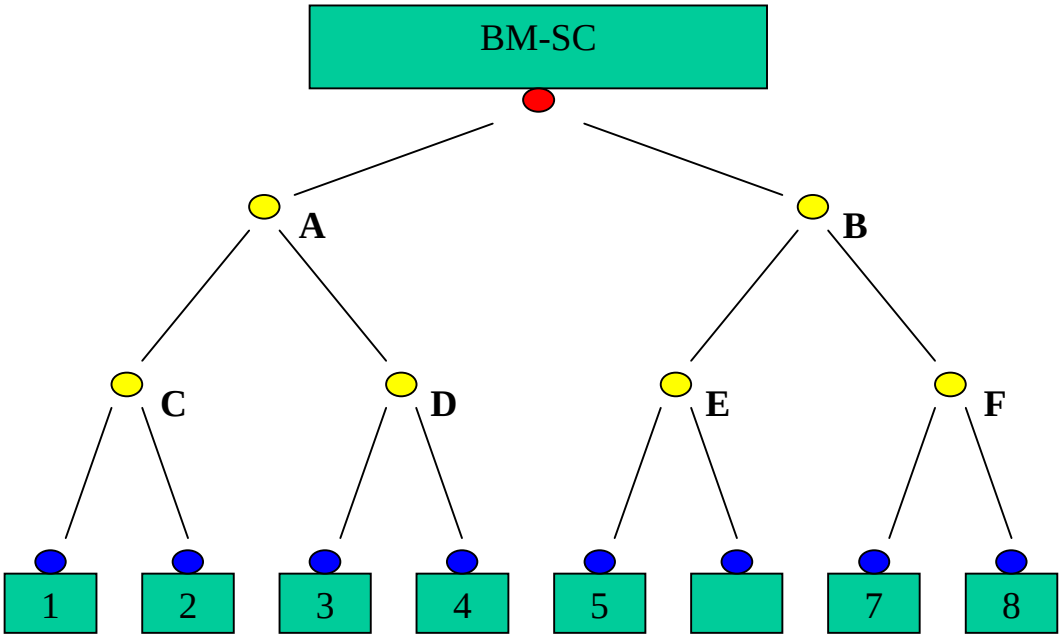

Figure 1. A Logical Key Hierarchy

Keys are represented by coloured circlesin the tree of Figure 1. The red circle represents the TEK, the yellow circles represent the LKH-keys, and the blue circles represent the keys shared only between the member and the MB-SC. The MB-SC has access to all keys in the tree including the TEK. Each member has access to all keys that lies above it in the path to the root, e.g. the leftmost member (member number 1) knows the keys TEK, A, C. Furthermore, each member shares a unique key with the MB-SC. It should be noted that the only vertices in the tree that corresponds to physical entities are the MB-SC and the members.

We describe the exclusion of members (i.e., re-keying of the remaining members) by an example. Assume that the leftmost member (member number 1) decides to leave the group. Then the MB-SC derives a new key TEK', which needs to be distributed to the rest of the group. Since the members {5, 6, 7, 8} all knows the key B, the new TEK is encrypted with the key B and sent in a multicast message to members {5, 6, 7, 8}.

The keys A and C are known to member 1, so they have to be replaced. The MB-SC derives new keys A' and C'.

Now, the MB-SC encrypts the new updated keys A' and TEK with key D and sends them in a multicast message to members 3 and 4.

Finally, A', C' and TEK' are encrypted with the key shared only by the MB-SC and member 2 and are sent to him in a single unicast message or the message is multicast.

The result of the above is that the keys A, C are changed to A' and C' respectively, and all members but member 1 have the new TEK'. There are several schemes based on this idea, and many optimisations can be made. The latest advancement in LKH is [2], which needs O(r) messages, where r is the number of members that are excluded from a group of size n. The constant hidden in the ordo notation is typically as low as 2 or 3. This in comparison to the unicast scheme, which requires n messages.

The authenticity of messages in a group setting is more complex than in a two-party setting. The reason being that there cannot be a shared key that is used for authentication, since a positive authentication only would imply that the message originated within the group. That is, it is not possible to determine which member of the group that sent the message. Basically one has to resort to signing the re-keying messages, which would put a heavy load on both terminals and servers.

Source Authentication

It is questionable whether source authentication of the re-keying messages is necessary in the MBMS scenario. If the access to the MBMS network is properly configured, there are mainly two points of attack: inserting packets in the air, or in the UTRAN.

To fake a re-keying message an attacker would first of all have to fake the sender IP-address of the MB-SC. If the GGSN is properly configured, it will filter out any packet arriving from the Internet that has the same IP-address as one of its MB-SC's. Similarly, if a UE tries to insert any IP-address other than its own in a packet and send it on the network, it is discarded by the GGSN. Consequently, the only possibility to send a faked re-keying message to a host is to insert it in the air (using an antenna) or hook up to the UTRAN. Right now both these seem difficult for an ordinary hacker, but the need for source authentication should be further studied.

Re-keying extension to MIKEY

In MIKEY, re-keying by using a unicast distribution can be done. This section focuses on extensions that may be need in MIKEY in order to make it possible to use MIKEY for multicast based re-keying (such as the use of LKH).

Basically, two things need to be specified. Firstly, how to distribute KEKs in the registration protocol, and secondly, how to build up the re-key message. It is possible to re-use much of what is already in MIKEY to achieve this. Basically, for the registration part, new key types must be registered in the key transport payload for the KEK/LKH key transport.

In the case of the re-keying, it is believed that the current payloads can be reused. However, as keys are protected with different keys depending on the aimed recipient in the key hierarchy, this must be specified (as the current use in MIKEY assumes the same key for protection of all keys).

The work needed to specify this is estimated to less than two months. However, as this will later require an IETF standardisation process, the total time needed may vary quite much. If the IETF 3GPP relation is used, it could be possible to put requirements on the timeframe for the overall process. In this case, two IETF meetings might be sufficient for the process (otherwise it is impossible to estimate the process time).

Conclusions

Of the two techniques presented in this discussion paper, only multicast based re-keying seems feasible for large groups. Unicast based re-keying induces unnecessary overhead, both in terms of bandwidth utilisation and workload on the key-server (connection establishment).

Furthermore, Section 4 shows that it is possible and what is required to extend MIKEY with the functionality of multicast based re-keying.

Proposal

We propose that re-keying in MBMS is done using MIKEY extended with LKH functionality.

References

[1] J. Arkko et. al, draft-ietf-msec-mikey-04.txt, September 2002

[2] D. Halevy, A. Shamir, The LSD broadcast encryption scheme, CRYPTO'02, Springer-Verlag Berlin Heidelberg, 2002

[3] D. Wallner et. al, Key management for Multicast: Issues and Architecture, IETF RFC 2627, June 1999