3GPP TSG SA WG3 Security — S3#21 **S3-010637**

27 - 30 November, 2001

Sophia Antipolis, France

Source: Siemens

Title: **SA distribution mechanism for the Ze interface**

Document for: Discussion / Decision

Agenda item: MAP security

Abstract

SA3 defined the Ze interface to distribute MAPsec SAs negotiated between different networks by their KAC entities. These are sent from the KACs to the NEs that use the MAPsec SAs to protect their MAP messages. The current discussion within SA3 focusses on a decision between a push and a pull model.

This contribution proposes an "extended" push model for MAPsec SA distribution.

1) Overview

As the recent discussion in 3GPP SA3 showed, neither a pure push-model nor a pure pull-model to distribute MAPsec SAs from the KAC to the NEs in the same network seems to be sufficient. In addition to the push model, functionality is needed to handle the case where a MAP-NE lost its current SAs or did miss recent updates. In addition to the pure pull model, functionality is required such that the KAC can tell the NEs about compromised SAs.

A detailed analysis of the different aspects of pure push/pull is provided in the joint Alcatel/Siemens contribution S3-0106xx ("Use of Push vs Pull Mechanisms in local SA distribution") as input to this meeting.

This contribution details the ‘***extended push’*** mechanism as to show its feasibility and is meant as a starting point for detailed discussion. The realization of the SA distribution protocol for the Ze-interface is not considered to be a task of SA3 but of CN4.
Before coming to details of an ‘extended push’ mechanism, the next paragraph list some requirements that are seen as essential for the realization of a protocol for the Ze-interface.

2) General requirements related to SA distribution over Ze

- The push mechanism requires the KAC to maintain active SAs with all other networks that MAP signalling is exchanged with. The KAC internally supervises the SA lifetime and performs automatic SA renewal with all other networks that MAP signalling is exchanged with.
- If new SAs are available, the KAC distributes these SAs to the NEs that require them, according to the KAC policy. The KAC must provide fresh SAs to its NEs for a specific network in time before the soft expiry time of the current active SA has been reached.
- The NEs must ensure that only Ze-messages sent by a KAC in the same security domain are accepted. Depending on the security domain policy, the NE may only accept Ze-messages from one or more specific KACs within the domain, out of the set of KACs in that domain.
- The KAC must ensure that only Ze-messages sent by a NE in the same security domain are accepted (Otherwise an attacker could, e.g. by sending false messages, force the KAC to overload the NE's)
- The KAC shall be able to know whether the MAP-NE needs only MAP-SA's for security domain internal communication or not.
- All messages over Ze should be integrity and confidentiality protected, as they contain sensitive data or trigger messages containing sensitive data, although integrity protection may be sufficient for trigger messages.

3) Proposed SA distribution procedures: 'Extended' Push mechanism

This section describes message flows for an extended push mechanism for MAPsec SA distribution over the Ze interface.

Case 1: Initial registration and SA distribution

To initiate communication with the KAC, each NE that requires MAPsec SAs initially registers with the KAC to obtain these SAs. The KAC then, according to the KAC policy, pushes the full set of active SAs that the NE requires (defined by network policy) to the NE, such that the NE receives all required SAs directly after registration. The "action" indicator is set to "REPLACE" to indicate that the NE shall use the SA_List as its new set of SAs, replacing all SAs that are currently in the NE's SADB.
The NE acknowledges the ZE_pushSA message after successfully installing the SAs in its SADB. Otherwise it responds with an error message (tbd).

```
KAC                                                     NE
       <----- ZE_register(NE_ID) -------------------
        ----- ZE_pushSA(action=REPLACE, SA_List) --->
       <----- ZE_ack(NE_ID, [error]) ---------------
```

Case 2: Subsequent SA distribution (normal case)

The KAC sends a ZE_pushSA message when a new SA is negotiated by the KAC to all NEs that require this SA. It is allowed to send several SAs as a list within a single ZE_pushSA message. The "action" indicator is set to "ADD" to indicate that the NE shall add the SA_List to its current SADB.
The NE acknowledges the ZE_pushSA message after successfully installing the SAs in its SADB. Otherwise it responds with an error message (tbd).

```
KAC                                                            NE
          ----- ZE_pushSA(action= ADD, SA_List) --->
         <----- ZE_ack(NE_ID, [error]) ------------
```

Note that the message format for the initial ZE_pushSA and the ZE_pushSA for SA updates is the same for both operations, since the parameter "SA_List" represents a single or a list of SAs.

Case 3: Handling of inconsistent NE states

In case of any event in a NE that leads to an inconsistent SA database in a NE, this NE sends a new register message, and receives the full set of active SAs from the KAC. The KAC, when receiving a new registration of an already registered NE, just updates any old registration with the new one.
Note that this step is considered to be relatively unlikely.

Case 4: SA revocation/removal

In case the revocation of an SA is required, the KAC sends a ZE_pushSA message identifying a list of the SAs to be removed to all NEs currently using them (the parameter SA_ID is tbd., but must uniquely identify the SA to be revoked. For example this can be the combination of SPI and destination PLMN_ID). A list of SAs may be added within the same ZE_pushSA (replacing the removed SAs). The "action" indicator is set to "REMOVE" to indicate that the NE shall delete the list of SAs identified in SA_ID_List from its current SADB, and the action "ADD" may be subsequently used to add the SAs contained in SA_List to its SADB.
The NE acknowledges the ZE_pushSA message after successfully removing the SAs from its SADB. Otherwise it responds with an error message (tbd).

```
KAC                                                                   NE
   ---------- ZE_pushSA(action=REMOVE, SA_ID_List,
                        action=ADD, SA_List) ---------------->
   <----------------- ZE_ack(NE_ID) -----------------------
```

Notes:
1) As the NEs in this push model do not have the option to request specific SAs, it is necessary to transmit the complete set of SAs from the KAC to the NE in cases 1 and 3.

2) The replacement of a compromised SA which needs to be revoked, described in case 4, could alternatively be done in two steps: by first sending a revoke operation, followed by an add operation as described in case 2.

3) The extended push mechanism, as described above, may be used to distribute policy information from the KAC to the NEs in a quite similar fashion, by using a SP_List, security policy list parameter instead of the SA_List parameter.

Calculation of the maximum amount of data to be transmitted on the ZE-interface (worst case):
A MAPsec SA, according to TS 33.200, consists of the following fields:
- MAP Encryption Algorithm identifier (MEA)
- MAP Encryption Key (MEK)
- MAP Integrity Algorithm identifier (MIA)
- MAP Integrity Key (MIK)

- Protection Profile Identifier (PPI)
- SA Lifetime

A reasonable assumption is that a single MAPsec SA will require less than 100 bytes
Thus, the maximum amount of data to be transfered by a ZE_pushSA(SA_List) message can be estimated as:
(2 SAs, one for each direction) x (<= 100 bytes per SA) x (<= 400 networks) <= 80kbyte

As this size is not considered to create problems like unacceptable transmission delays, no requirement for supporting individual SA requests by the NEs can be seen.

4) Conclusion:

This contribution shows a feasible solution of the extended push model for SA-distribution on the Ze-Interface.
The general requirements on the Ze-interface shall be used for the selection and definition of the protocols and the described ‘extended push’ mechanism shall be used as a starting point for the protocol definition.