3GPP TSG SA WG3 Security — S3#35 **S3-040775**
5 – 8 October 2004
Malta

Source: **Gemplus, Oberthur, Axalto**

Title: **GBA_U: Alternatives for GBA_U derivations**

Document for: **Discussion and decision**

Agenda Item:

1. Introduction

In case of the storage of Ks_ext on the UICC, it was proposed at SA3#34 to optimise the GBA_U bootstrapping procedure by removing at least one key derivation procedure. Ks_int and Ks_ext could be replaced with a single key Ks. This contribution proposes a study of the different alternatives.

2. Current status of GBA-U key derivation

The current TS 33.220 [1] proposes the following key derivations for GBA-U:

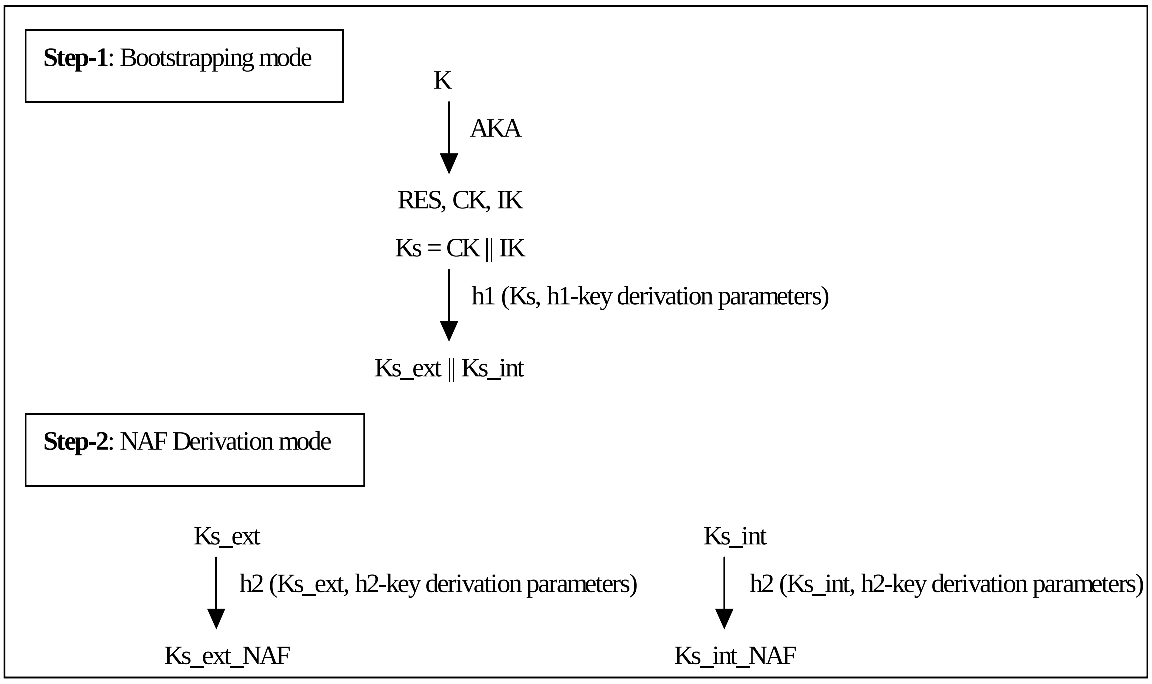

Step-1: Bootstrapping mode
K
AKA
RES, CK, IK
Ks = CK || IK
h1 (Ks, h1-key derivation parameters)
Ks_ext || Ks_int
Step-2: NAF Derivation mode
Ks_ext
h2 (Ks_ext, h2-key derivation parameters)
Ks_ext_NAF
Ks_int
h2 (Ks_int, h2-key derivation parameters)
Ks_int_NAF

Figure 1: Current derivation methods to compute Ks_ext_NAF and Ks_int_NAF

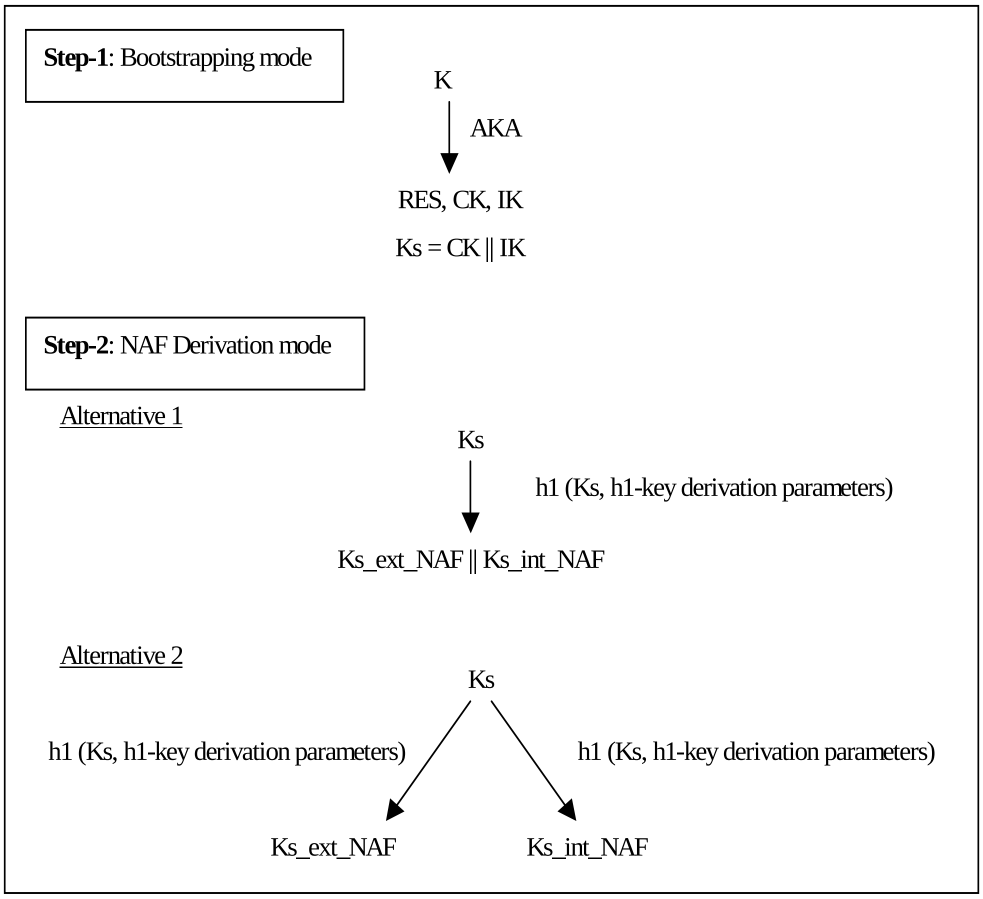
Ks_ext and Ks_int are 128-bit keys
The definition of the key derivation function is left to ETSI SAGE.

3. Alternatives

In case of Ks_ext stored on the UICC, it was proposed at SA3#34 to study the possibility to replace Ks_int and Ks_ext with a single key Ks since Ks_ext is no longer sent to the ME, the UICC only sends Ks_ext_NAF.

3.1. Description

The following schemes are possible:



Step-1: Bootstrapping mode
K
AKA
RES, CK, IK
Ks = CK || IK
Step-2: NAF Derivation mode
Alternative 1
Ks
h1 (Ks, h1-key derivation parameters)
Ks_ext_NAF || Ks_int_NAF
Alternative 2
Ks
h1 (Ks, h1-key derivation parameters)
h1 (Ks, h1-key derivation parameters)
Ks_ext_NAF
Ks_int_NAF

Figure 2: Alternatives for Ks_xx_NAF derivation from Ks

Step-1
There is no longer h1 key derivation function in the bootstrapping mode. The single key Ks is the concatenation of CK and IK.

Step-2
Alternative-1:
The output size of h1 key derivation function is at least two times the length of Ks_xx_NAF key.

Alternative-2:
The computation of Ks_ext_NAF and Ks_int_NAF requires two executions of the key derivation function h1, one derivation for the external key Ks_ext_NAF and one derivation for the internal key Ks_int_NAF.
The h1 key derivation function may either be the same function, in which case the differentiation between external and internal keys will happen via different parameters, or be a set of two different functions, h1_ext and h1_int.
The output size of the h1 key derivation function is at least the length of Ks_xx_NAF key.

3.2. Choice of the key derivation function

Requirement to fulfill

The solution should fulfill the following requirement: the key size of Ks_xx_NAF with GBA_U shall be the same than the key size of Ks_NAF with GBA_ME .

Ks_NAF and Ks_xx_NAF key size

The choice of the key derivation function depends on the key length of Ks_NAF.

At the moment, the proposed key length for Ks_NAF (GBA_ME) is 256 bits according to the LSs sent by SA3 to SAGE ([1] and [2]). This contribution also studies the possibility to have Ks_NAF and Ks_xx_NAF with a length of 128 bits.
The reason to propose 128-bit keys is due to the fact that Ks_NAF and Ks_xx_NAF are derived from K, which is a 128-bit key. The entropy of a derived key cannot be higher than the entropy of the initial secret.
We kindly ask SA3 to restate the reasons to choose Ks_NAF of length 256 bits.

The following table shows the features of the h1 key derivation function (KDF) according to the alternative and the length required for the NAF-specific key:

	GBA_ME Step 2: NAF derivation mode	**GBA_U Step-2**: NAF Derivation mode	
	KDF h1	**Alternative 1:** KDF h1	**Alternative 2:** KDF h1
NAF-specific key: **256-bit key**	Output size: at least 256 bits (e.g. HMAC-SHA-256) The KDF is executed once	Output size: at least 512 bits The KDF is executed once	Output size: at least 256 bits (e.g. HMAC-SHA-256) The KDF is executed twice
NAF-specific key: **128-bit key**	Output size: at least 128 bits The KDF is executed once	Output size: at least 256 bits (e.g. HMAC-SHA-256) The KDF is executed once	Output size: at least 128 bits The KDF is executed twice

Figure 1: Description of the key derivation function according to the alternatives

128-bit NAF-specific key:
The most interesting solution seems to be the Alternative-1. The output of the h1 key derivation function would correspond to the concatenation of Ks_ext_NAF and Ks_int_NAF. The use of HMAC-SHA-256 could be kept.

256-bit NAF-specific key:
The choice of the alternative depends on the complexity to implement the key derivation function and the time processing to execute the command.

3.3. Benefits of the alternatives

The use of Ks reduces the bootstrapping time since in the Bootstrapping mode the UICC would have to perform only the concatenation of CK and IK whatever the alternative selected for the NAF Derivation mode. It also reduces the implementation complexity in the BSF, as the GBA_U procedure will be similar to the GBA_ME procedure at least for the Bootstrapping mode (Step-1).

So, it will lead to better performance in the UICC and the BSF.

4. Conclusion

The proposal for replacing Ks_int and Ks_ext with a single key Ks reduces the derivation complexity and the processing time of the GBA_U procedure in UICC and BSF sides.

So, we recommend the use of a single key Ks. The choice of the key length, the alternative and the key derivation function for the NAF Derivation mode is left to SA3 and ETSI SAGE.

A CR [3] implements one of the alternatives. This CR could be updated during SA3#35 meeting to reflect SA3's decision.

5. References

[1] TD S3-040162, "LS on key derivation for the Generic Bootstrapping Architecture", SA3#32

[2] TD S3-040448, "Response to LS (S3-040268) on key derivation for the Generic Bootstrapping Architecture", SA3#33

[3] TD S3-040xxx, "CR: Optimization of the GBA_U key derivation procedure", Gemplus, Axalto, Oberthur SA3#35