ETSI TS 133 223 V8.2.0 (2009-01)

Technical Specification

Digital cellular telecommunications system (Phase 2+);
Universal Mobile Telecommunications System (UMTS);
LTE;
Generic Authentication Architecture (GAA);
Generic Bootstrapping Architecture (GBA) Push function
(3GPP TS 33.223 version 8.2.0 Release 8)

Reference
DTS/TSGS-0333223v820

Keywords
GSM, LTE, SECURITY, UMTS

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Specification (TS) has been produced by ETSI 3rd Generation Partnership Project (3GPP).

The present document may refer to technical specifications or reports using their 3GPP identities, UMTS identities or GSM identities. These should be interpreted as being references to the corresponding ETSI deliverables.

The cross reference between GSM, UMTS, 3GPP and ETSI identities can be found under http://webapp.etsi.org/key/queryform.asp.

Contents

Foreword

This Technical Specification has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

Introduction

3GPP defined the Generic Authentication Architecture (GAA). The adoption of GAA by other standardization bodies showed that some services can not make the assumption that the User Equipment (UE) has always the possibility to connect to the Bootstrapping Server Function (BSF) or that the UE for different reasons has not performed a bootstrapping procedure directly with the BSF. Hence, this specification introduces and specifies a GBA Push Function.

1 Scope

The present document specifies a Push Function as a functional add-on for the Generic Authentication Architecture (GAA) [1].

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TS 33.220: "Generic Authentication Architecture (GAA); Generic bootstrapping architecture".

[2] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[3] 3GPP TS 33.210: "3G Security; Network Domain Security; IP network layer security".

[4] IETF RFC 2246 (1999): "The TLS Protocol Version 1".

[5] Void.

[6] 3GPP TS 33.102: "3G Security; Security architecture".

[7] FIPS PUB 180-2 (2002): "Secure Hash Standard".

[8] IETF RFC 2104 (1997): "HMAC: Keyed-Hashing for Message Authentication".

[9] ISO/IEC 10118-3:2004: "Information Technology – Security techniques – Hash-functions – Part 3: Dedicated hash-functions".

[10] NIST Special Publication 800-38A: "Recommendation for Block Cipher Modes of Operation"

[11] FIPS PUB 197: "Advanced Encryption Standard"

[12] RFC 1305: "Network Time Protocol (Version 3)"

[13] 3GPP TS 33.222 "Access to network application functions using Hypertext Transfer Protocol over Transport Layer Security (HTTPS)

3 Definitions, symbols and abbreviations

3.1 Definitions

For the purposes of the present document, the terms and definitions given in TR 21.905 [2], TS 33.220 [1] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in TR 21.905 [2].

AUTN(*): In GBA context, GBA_ME relies on AUTN value to verify that the authentication vector is from an authorised network, while GBA_U relies on AUTN* to perform network authentication as described in [1]. AUTN(*) is used to refer both to AUTN and AUTN*.

Disposable-Ks model: The keying model used in GBA-push. Only one NAF-key is generated per Ks and the Ks cannot be reused.

GBA_U aware UICC: A UICC which supports GBA_U which means that the Ks will never leave the UICC.

GBA-Push-Info: GBA-Push-Info contains data relevant for key derivation in GBA Push. GBA-Push_Info is sent via the Upa-reference point from the NAF to the UE.

NAF_Id: The FQDN of the NAF, concatenated with the Ua security protocol identifier,

NAF-key: A NAF-key derived from Ks. It can be used to refer to Ks_(int/ext)_NAF or Ks_NAF.

NAF SA: A security association between a NAF and a UE based on a NAF-key.

Push-message: This is a message that is sent on a Ua-reference point from the NAF to the UE and has applied GBA keys that were bootstrapped via the Upa-reference point.

Push-NAF: A NAF authorized for using GBA-Push.

UE_Trp: The transport address used for delivery of GPI to the UE.

3.2 Abbreviations

For the purposes of the present document, the abbreviations given in TR 21.905 [2] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in TR 21.905 [2].

BSF	Bootstrapping Server Function
B-TID	Bootstrapping Transaction Identifier
FQDN	Fully Qualified Domain Name
GAA	Generic Authentication Architecture
GBA	Generic Bootstrapping Architecture
GBA_ME	ME-based GBA
GBA_U	GBA with UICC-based enhancements
GPI	GBA Push Info
GUSS	GBA User Security Settings
HSS	Home Subscriber Server
Ks_NAF	NAF-key in GBA_ME mode
Ks_int_NAF	UICC internal NAF-key in GBA_U
Ks_ext_NAF	UICC external NAF-key in GBA_U
ME	Mobile Equipment
NAF	Network Application Function
SA	Security Association
UE	User Equipment
USS	User Security Setting

4 GBA Push Architecture

4.1 Introduction

4.1.1 General

GBA-push is a mechanism to bootstrap the security between a NAF and a UE, without forcing the UE to contact the BSF to initiate the bootstrapping. GBA-Push is closely related to and builds upon GBA as specified in TS 33.220 [1]. GBA-Push is aimed for both GBA_U and GBA_ME environments.

In an accompanying specification, TS 33.224 [5], a message format for secure Push-messages, the Generic Push Layer, is defined.

4.1.2 GBA-Push system overview

The system overview in this clause gives a high level description of the general ideas behind the GBA-Push system solution and the features it offers.

The generic use case considered is that a NAF initiate's establishment of a shared Security Association (SA), a NAF SA, between itself and a UE. This is done by the NAF pushing all information, the so called GBA-Push-Info (GPI), needed for the UE to set-up the SA. The key in this SA is a NAF-key and the GPI is requested from the BSF. The NAF-key is generated as defined in GBA, TS 33.220 [1].

After the NAF SA establishment, the NAF can send protected Push-messages to the UE. If a return channel exists and if defined by the Ua application, the UE can also use the established SA to protect response messages to the initiating NAF How the NAF SA is used is out of scope for this specification. The NAF SA is identified by downlink and uplink SA identifiers.

GBA-Push is aimed for both GBA_U and GBA_ME environments. To only establish an external NAF-key with GBA-Push, the ME-based functionality, GBA_ME, should be used. GBA-Push based on GBA_U will establish both an internal and external NAF-key.

GBA-Push utilizes a so called Disposable-Ks model. In the Disposable-Ks model, a Ks is only used once to derive a single set of NAF-keys (and other keying material used to protect the GPI during transport). After the NAF-key derivation, the Ks is erased or its further usage is denied. A new GBA-Push operation will be needed whenever a new set of NAF-keys for the same or another NAF is needed.

NOTE 1: A generated NAF-key can be used to protect multiple Push-messages from the NAF to the UE. NAF-keys from different NAFs can coexist.

With the Disposable-Ks model, existing NAF-keys established as specified in TS 33.220 [1] or by GBA-Push will be unaffected. GBA_ME based GBA-Push will not interact with GBA_U but a GBA_U based GBA Push will invalidate an existing Ks on the UICC.

NOTE 2: TS 33.220 [1] specifies that an existing Ks on the UICC will be overwritten when a new GBA_U Ks-generation procedure is executed. The ME may of course trigger a new bootstrap procedure immediately after the GBA-Push operation to avoid delays and certain synch problems when the UE operates GBA according to TS33.220 [1].

The transport of GPI from a NAF to a UE is not standardized.

NOTE 3: Examples of possible transport methods are SMS, MMS, SIP MESSAGE, UDP or broadcast. For the transport of GPI to UEs, a NAF needs to know the message transport addresses to use for the chosen transport method. For SMS and MMS the transport address is the MSISDN, for SIP MESSAGE it is an IMPU and for UDP an IP-address. For broadcast delivery the UE transport addresses could be any public identity associated with a UE or an identity agreed between the NAF and the UE.

Resending of messages is a standard method to get 'reliability' for delivery over unreliable channels like e.g. SMS or broadcast. Hence the GBA-Push shall allow that GPI is retransmitted several times including cases when it is sent every time a payload is pushed to the UE. Thus the system shall handle retransmissions of GPI efficiently.

The NAF SA defined by the GPI, is based on the use of a particular UICC (USIM/ISIM) application. Sometimes the transport method / address indicate to the UE which UICC application to use but in other cases it has to be explicitly signaled. If MSISDN is used as delivery address, then the USIM associated with that MSISDN should be used. This is so because a SMS will only reach the UE when the USIM corresponding to the MSISDN is active in the UE. When an IMPU is used as destination address, the corresponding ISIM should be used. For UDP and broadcast the USIM/ISIM application to use has to be indicated in the GPI or be agreed upon out of band.

To protect user privacy, parts of the GPI shall be confidentiality protected, in particular the identity of the initiating NAF when broadcast transport is used. For unlinkability between NAF to UE and UE to NAF messages, a separate SA identity for UE to NAF security shall be assigned by the NAF and be included in the confidentiality protected part of the GPI. To help prevent serious effects of DoS attacks and thwart some NAF misuse of GBA-Push, the GPI also needs to be integrity protected. The integrity protection of GPI will also prevent that incorrect GBA Push security associations

are accepted by the UE as it will detect transmission errors. The keys for confidentiality and integrity protection are derived from the Ks defined by the GPI.

4.2 GBA Push Architecture

4.2.1 Description and Rationale

The GBA Push functionality builds on the architecture and functionality provided by TS 33.220 [1]. The main difference from TS 33.220 [1] is the definition of a new reference point between the BSF and the NAF, as indicated in figure 4.2-1, which is a modified version of figure 4.1 in TS 33.220 [1].

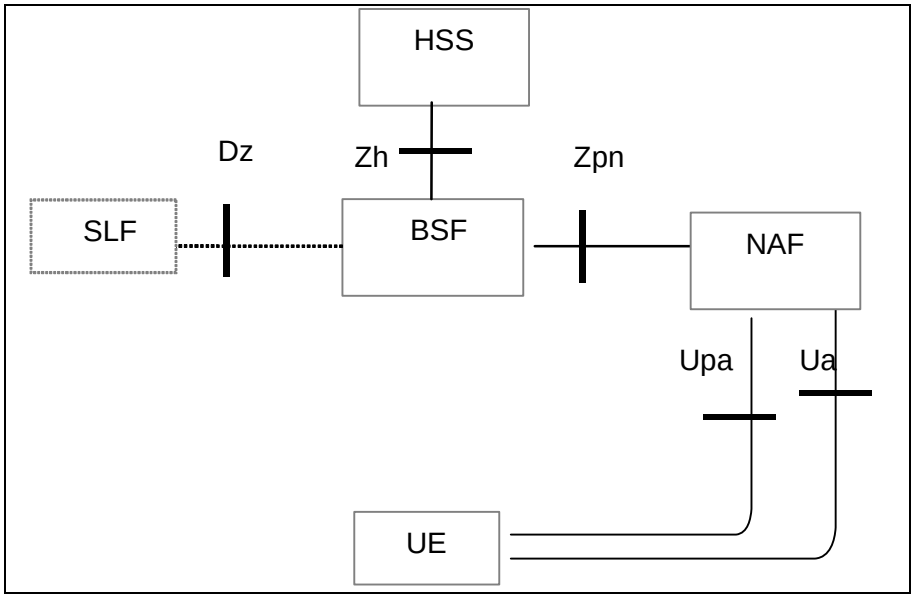

Figure 4.2-1: Simple network model for pushed bootstrapping via NAF

The GBA Push architecture outlined in figure 4.2-1 is based on the following rationales:

- The Ua reference point protection shall be unaffected i.e. it should not make any difference for Ua-protocols whether the GBA-keys used for protection are UE-initiated or push-initiated.
- In viewpoint of the BSF, the NAF is still the initiating entity of a key retrieval, but now in situations where the NAF has no B-TID (but the UE may have a valid GBA session). A Zpn reference point is introduced, based on the Zn-reference point protocols defined by TS 33.220 [1].
- A new reference point Upa is introduced between the NAF and the UE. All messages over Upa are network initiated. Upa defines the GBA-Push-Info.
- The NAF receives the GBA-Push-Info intended for the UE from the BSF over the Zpn reference point and forwards it over Upa.

4.2.2 GBA-Push keying model

The Disposable-Ks model is the keying model used in GBA-Push. In the Disposable-Ks model, a Ks is only used once to derive a single set of NAF-keys (and other keying material used to protect the GPI during transport, see clause 5.3). After the NAF-key derivation, the Ks is erased or its further use is denied implicitly, which means that there will be no generally usable Ks established.

To only establish an external NAF-key with GBA-Push, GBA_ME can and should always be used. This functionality does not require a GBA_U aware UICC. GBA-Push based on GBA_U will establish both an internal and an external NAF-key. NAF-keys are derived as specified in TS 33.220 [1].

In GBA_ME based GBA-Push bootstrapping, a Ks, generated by a bootstrapping according to TS 33.220 [1], will be unaffected.

In GBA_U based GBA Push bootstrapping, a GBA_U Ks generated by bootstrapping according to TS 33.220 [1] will be invalidated. A new GBA_U Ks needs to be established using normal GBA if an application requires GBA_U NAF-keys after GBA_U based GBA-Push bootstrapping. Applications can continue using NAF-keys derived from such an invalidated Ks, i.e. applications already using NAF-keys are unaffected of the GBA-Push bootstrapping run.

GBA-Push only supports generation of so called NAF SAs, shared by a UE and a NAF. A NAF SA contains a NAF-key, key life-time and other information as defined in clause 5.2.3.

4.3 GBA Push Requirements

4.3.1 General GBA Push Requirements

The following general requirements are applicable to enable GBA Push:

- A network entity, a so called Push NAF, shall be able to securely trigger the generation of a NAF SA between itself and a UE.
- A Push-NAF shall be able to use channels with deferred delivery of messages when triggering the generation of a NAF SA.
- A Push-NAF shall be able to use public identities when referencing a UE in a request towards the BSF.
- When a public identifier is used for GBA push it shall correspond uniquely to a single private identity..
- ME based GBA Push shall be used when only ME based NAF keys are needed, i.e. Ks is established in the ME. UICC based GBA Push shall be used only when UE contains a GBA aware UICC (GBA_U), and UICC and ME based NAF keys are needed, i.e. Ks is established in the UICC.
- The generation of the NAF SA in the UE is triggered by the reception of a message pushed to the UE from the Push-NAF.
- The UE should not have to contact any network entity to be able to correctly generate the NAF SA.
- The UE and the NAF shall be able to use bootstrapped NAF-keys on Ua reference point independent on whether the bootstrapping has been performed via Ub or Upa reference point.

NOTE: When a GBA-push mechanism is used to create a NAF SA between the UE and the NAF it shall not restrict the NAF to use the derived security association for network initiated protocols only. Analogously, the fact that UE initiated GBA was used shall not restrict a NAF to use the derived security association for UE-initiated protocols only (Ua reference point).

- The mechanism to generate keys for confidentially and integrity protection of GPI shall be based on GBA-keys in order to avoid pre-configuration of keys.
- The NAF shall be unable to obtain or generate the keys that protect GPI.

4.3.2 Requirements on HSS

The requirements for HSS are in TS 33.220 [1].

4.3.3 Requirements on BSF

In addition to the BSF requirements in clause 4.2.1 of TS 33.220 [1] following requirements apply:

- The BSF shall be able to find the private identity corresponding to a public identity.
- The BSF shall index existing Ks's based on private user identity.
- The BSF shall generate GPI based on a fresh Ks.
- The BSF shall integrity protect the GPI.

- The BSF shall confidentiality protect certain fields in the GPI. The fields that shall be confidentially protected are given in clause 5.2.1.

4.3.4 Requirements on UE

In addition to the UE requirements in clause 4.2.4 of TS 33.220 [1] the following requirements apply:

- The UE shall be able to store and handle NAF SAs.
- A GBA aware ME, TS 33.220 [1], shall support GBA-Push as well as GBA_U and GBA-ME.
- The UE may implement an authorization mechanism to authorize incoming GBA Push messages.

NOTE: The GBA Push message authorization mechanism can be based on white or black lists of FQDN names of the Push-NAFs.

4.3.5 Requirements on Reference Point Upa

The requirements for reference point Upa are:

- The UE shall be able to validate that the GPI comes from an authorized source (BSF) based on AKA

NOTE 1: The Push-NAF is indirectly authenticated by its knowledge of Ks(_ext/int)_NAF (i.e. BSF has authenticated the NAF).

- The UE shall be able to determine the UICC (USIM/ISIM) application used for bootstrapping.
- The NAF and the UE shall be able to establish a shared NAF SA.
- The NAF shall be able to send NAF SA identifier information.
- the BSF shall be able to indicate to the UE the lifetime of the key material. The key lifetime sent by the BSF over Upa shall indicate the expiry time of the key.

NOTE 2: The requirements for the Upa reference point are based on the requirements of the Ub reference point c.f. TS 33.220 [1].

4.3.6 Requirements on Reference Point Zh

The requirements for reference point Zh are in TS 33.220 [1].

4.3.7 Requirements on Reference Point Zpn and Zpn'

The requirements for reference point Zpn are:

- Mutual authentication, confidentiality and integrity shall be provided.
- If the BSF and the NAF are located within the same operator's network, the DIAMETER based Zpn reference point shall be secured according to NDS/IP, TS 33.210 [3].
- If the BSF and the NAF are located in different operators' networks, the DIAMETER based Zpn' reference point between the Zn-Proxy and the BSF shall be secured using TLS as specified in RFC 2246 [4].

NOTE 1: Annex E of TS 33.220 [1] specifies the TLS profile that shall be applied.

- A Web Services based Zpn/Zpn' reference point shall be secured using TLS as specified in RFC 2246 [4];

NOTE 2: Annex E of TS 33.220 [1] specifies the TLS profile that shall be applied.

- The BSF shall verify that the requesting NAF is authorised to obtain the key material or the key material and the requested USS.

- The NAF shall be able to send a key material request to the BSF, containing NAF's public hostname corresponding to the use over Upa reference point. The BSF shall be able to verify that a NAF is authorized to use this hostname, i.e. the FQDN seen by UE on Upa reference point.

NOTE 3: This requirement is a modified requirement from [1] that has been adapted for the GBA Push purpose.

NOTE 3a: Due to the fact that the UE may be unable to verify the pNAF FQDN, it is important to strictly check the pNAF FQDN-name in the network in the Zpn-proxy. A too loose checking of the pNAF FQDN name e.g. by verification of only part of the FQDN, may give rise to misuse by pNAFs.

- The BSF shall be able to send the requested key material to the NAF.

- The NAF shall be able to get a selected set of application-specific USSs from the BSF, depending on the policy of the BSF and the application indicated in the request from the NAF over Zpn.

- The NAF shall be able to indicate to the BSF the single application or several applications it requires USSs for.

NOTE 4: If some application needs only a subset of an application-specific USS, e.g. only one IMPU, the NAF selects this subset from the complete set of USS sent from BSF.

- The BSF shall be able to be configured on a per NAF or per application basis.

- Whether private subscriber identity, i.e. IMPI, may be sent to the NAF.

- Whether a particular USS may be sent to a NAF.

- If a NAF requests USSs from the BSF and they are not present in subscriber's GUSS, it shall not cause an error, provided the conditions of the local policy of the BSF are fulfilled. The BSF shall then send only the requested and found USSs to the NAF.

- It shall be possible to configure a local policy as follows: BSF may require one or more application-specific USS to be present in a particular subscriber's GUSS for a particular requesting NAF, and to reject the request from the NAF in case the conditions are not fulfilled. In order to satisfy this local policy, it is not required that the NAF requests the USSs over the Zpn reference point, which the BSF requires to be present in the GUSS, rather it is sufficient that the BSF checks the presence of the USSs locally. It shall also be possible to configure the BSF in such a way that no USS is required for the requesting NAF.

NOTE 5: For more information on the local policy usage, see Annex J of TS 33.220 [1].

- The NAF shall be able to request the life-time that a NAF SA should have. The key lifetime sent by the BSF over Zpn shall indicate the expiry time of the key.

NOTE 6: This does not preclude a NAF to refresh the NAF SA before the expiry time according to the NAF's local policy.

NOTE 7: If one or more of the USSs that have been delivered to the NAF has been updated in subscriber's GUSS in the HSS, this change is propagated to the NAF the next time it fetches the USS from the BSF over Zpn reference point (provided that the BSF has updated subscriber's GUSS from the HSS over Zh reference point).

- NAF shall be able to indicate to BSF the protocol identifier of Ua security protocol for which it requires the key material (cf. Annex H of TS 33.220 [1]).

- The NAF shall be able to indicate the user identity to the BSF. Both public and private identities shall be allowed.

NOTE 8: The requirements for reference point Zpn are based on the Zn-reference point requirements as described in TS 33.220 [1].

- The NAF shall be able to indicate whether GBA_ME or GBA_U shall be used.

4.3.8 Requirements on Zn-Proxy

In the case that push NAF is operated in a network other than the home network, this visited NAF shall use a Zn-proxy of the NAF's network to communicate with the subscriber's BSF (i.e. home BSF). The requirements for the Zn proxy are described in TS 33.220 [1].

4.3.9 Requirements on Reference Point Ua

The requirements for reference point Ua are as in TS 33.220 [1] with the following addition:

- It shall be possible to use SA identifiers in the uplink that are unlinkable with the push message establishing the used NAF SA.

4.3.10 Requirements on NAF SA identifiers

- The downlink NAF SA identifier shall be unique within the UE and uniquely identify that it references a NAF SA for a particular NAF_Id.

- The uplink NAF SA identifier shall be unique within the NAF and uniquely identify that it references a NAF SA for a particular UE and Ua security protocol identity.

4.3.11 Requirements on Reference Point Dz

This interface between BSF and SLF is used to retrieve the address of the HSS and the requirements are the same as described in TS 33.220 [1]. This interface is not required in a single HSS environment.

5 GBA Push Function

5.1 GBA Push Message Flow and Processing

5.1.1 GBA Push Message Flow

Figure 5.1-1 outlines the message flow for the case, where the NAF wants to send data to the UE, but has no valid NAF-key available i.e. no Ks(_int/ext)_NAF available The reason that the NAF has to initiate NAF SA establishment can be that the UE may be unable to perform a bootstrapping procedure directly with the BSF or that the UE should not perform a bootstrapping procedure directly with the BSF

NOTE 1: An example use case when the UE is unable to perform a bootstrapping procedure is in a broadcast scenario.

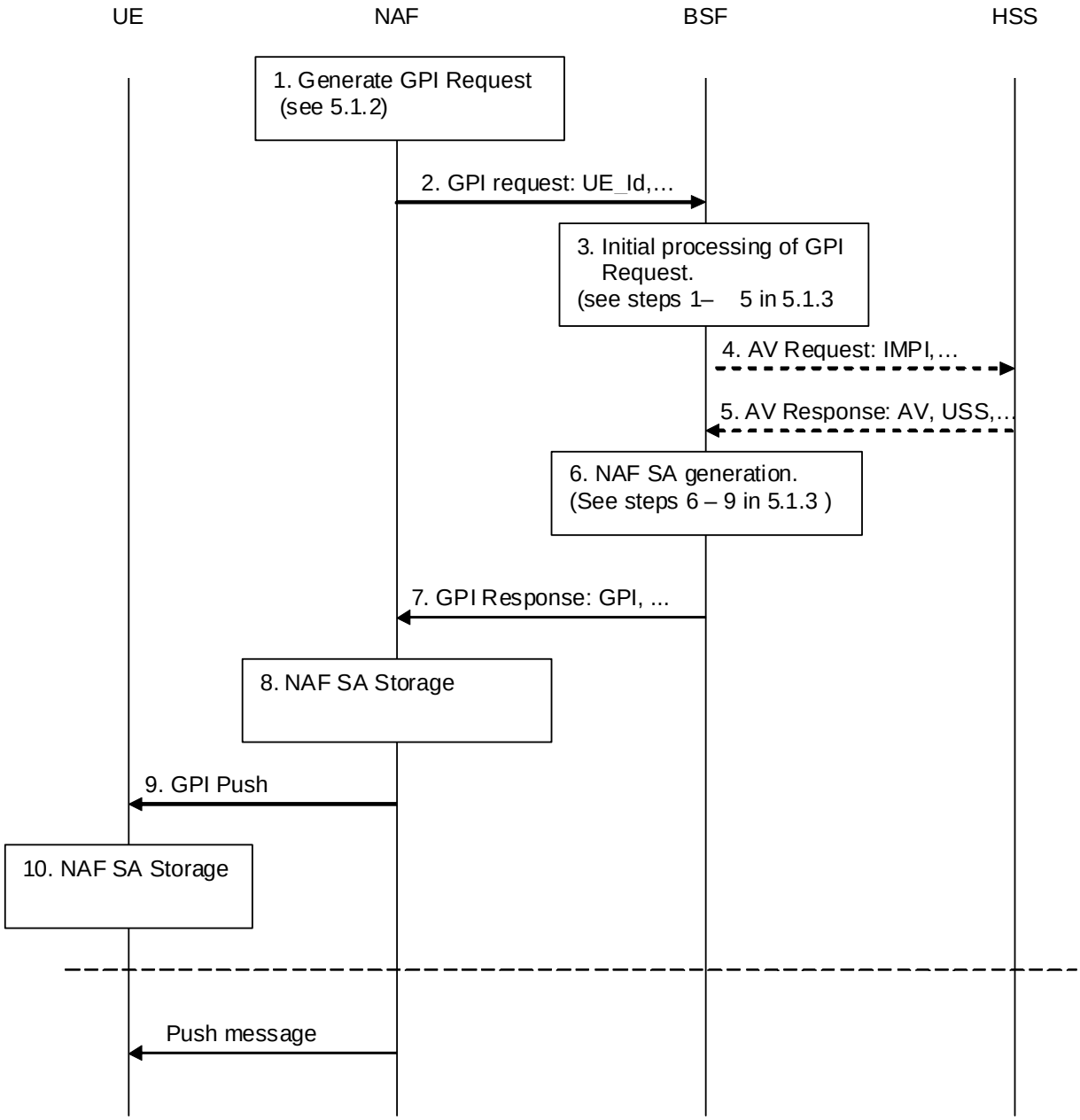

Figure 5.1-1: High level message flow description for bootstrapping through the NAF

A precondition for use of GBA-Push is that the UE is registered with the Push-NAF for the intended service. Annex B describes information that the Push-NAF must register to be able to deliver the push service and the information that has to be agreed between the UE and the Push-NAF.

Processing and message flow:

1. A NAF needs to establish a shared NAF SA with a UE which is registered for Push services. It knows the identity of the subscriber. The Push-NAF performs the processing described in clause 5.1.2 and generates the GPI Request.

2. The NAF sends the GPI Request to the BSF.

3. Upon receiving the request from the NAF, the BSF performs the processing steps 1 to 5 described in clause 5.1.3.

4. The BSF fetches a new AV and subscriber's GUSS from the HSS. The GUSS contains subscriber security related information e.g. UICC GBA awareness and USS elements.

5 The HSS sends the AV and the GUSS to the BSF.

6. When the BSF receives the AV Response from the HSS, it performs the processing steps 6 to 9 described in clause 5.1.3.

7. The BSF sends the GPI Response to the NAF.

8. The NAF stores the received information together with other user information in a NAF SA, see clause 5.2.3.

9. The NAF then forwards the GPI to the UE over Upa using the selected transport mechanism and the given transport address.

10. When the UE receives the message containing the GPI, it processes the GPI as defined in clause 5.1.4 and stores the corresponding NAF SA(s)

The NAF is now ready to use the established NAF SA.

5.1.2 NAF processing before issuing GPI request

The NAF reads its available data associated with the user and the application for which the NAF SA shall be established. The NAF then determines the Ua security protocol identifier to use in the request to the BSF. It also determines the required life-time of the NAF SA. The NAF then generates the GPI request containing the parameters as given in table 5.1.2.1

Table 5.1.2.1: Parameters in NAF GPI request

Parameter name	Description	Notes
UE_Id	UE identifier	This may be a private or a public identifier.
UE_Id_Type	Indicator if UE_Id is a public or private identity	This information is needed by the BSF to correctly trigger the Public to Private Id resolution towards a HSS/HLR
App_Lbl	Identifier for UICC application to use	This variable may be left empty if the UICC application to use is evident from context or agreement.
NAF_Id	Concatenation of NAF FQDN and Ua security protocol Id	Defined in TS 33.220 [1]
P-TID	NAF SA identifier	To be used by UE when responding to NAF. The identifier is included only to enable that it is confidentiality protected in the GPI. See also clause 5.2.2 and 5.2.4.
U/M	Indicator for use of GBA_ME or GBA_U	
Key_LT	Requested NAF-Key life time	
Priv_Id	Indicates request for private user identity	Private user identity is IMSI/IMPI for the selected UICC application (USIM/ISIM)
GSID_List	GSIDs of USS request information	

5.1.3 BSF processing of NAF GPI request

When the BSF receives the GPI request from the NAF it performs the following processing steps:

1. The BSF checks that the NAF is authorized to use the NAF_Id provided in the GPI request. If it is not, an error message is generated and the processing is terminated.

The BSF checks that the requested Key_LT in the GPI request is less than the allowed max value in the system. If the value is greater than the max value an error message is generated and the processing is terminated.

2. If the UE_Id is a public identity the BSF (by unspecified methods) retrieves the corresponding private identity (i.e. IMPI or IMSI).

3. If needed, the BSF retrieves the HSS address for the given UE using the SLF.

4. The BSF requests an AV, and subscriber's GUSS from the HSS.

5. The BSF checks if GBA_ME or GBA_U is requested by the NAF. If GBA_U is requested the BSF checks that this is compatible with the GBA awareness of the UICC of the GUSS. If it is not, an error message is generated and the processing is terminated.

   The BSF may use USS for policy management and key selection indication as described in TS 33.220 [1].
   If GBA_U is requested the BSF queries its database to find out if the private UE_Id is registered and if a valid Ks already exists. If a valid Ks exists the BSF shall invalidate this Ks.

6. The BSF generates the requested NAF-key(s) according to provided NAF_Id.

7. The BSF generates the GPI. The parameters of the GPI are defined in clause 5.2.1. The generation of the GPI includes calculation of the GPI MAC and performing confidentiality protection on parts of the GPI. GPI protection is described in clause 5.3.

8. The BSF sends its response to the NAF, and deletes the Ks used. The GPI response is defined in table 5.1.3.1.

Table 5.1.3.1: Parameters in GPI response

Parameter name	Description	Notes
GPI	GPI	GPI information is defined clause 5.2.1
Ks_NAF / Ks_ext_NAF	External NAF-key	Ks_NAF is generated in GBA_ME based GBA-Push Ks_ext_NAF is generated in GBA_U based GBA_Push
Ks_int_NAF	UICC internal NAF-key	Ks_int_NAF is generated in GBA_U based GBA_Push
Key_LT	NAF-Key life time	
UE_Priv_Id	Private user identity (IMSI/IMPI) for used UE_Id	Only returned if requested and public user identity was used in GPI request.
USS	USS information	

5.1.4 UE processing of GPI

When the UE receives a GPI it performs the following steps.

1. UE receives GPI. The parameters of the GPI are defined in clause 5.2.1

2. If the App_Lbl in the GPI is undefined, the UE determines the UICC application to use from used delivery channel of the GPI (e.g. SMS, MMS, SIP Message, etc) or from other context information.

3. UE checks if it has received the same GPI earlier.

   a. If the GPI corresponds to an already existing NAF SA, then the GPI is silently dropped and the GPI processing terminated.

   b. If the GPI corresponds to an incomplete NAF SA, the Ks indicated by GPI is activated and processing continues from step 6 (step 7 describes how an incomplete SA may appear).

NOTE 1: To handle retransmissions efficiently the UE should only invoke a UICC application after checking that the GPI does not correspond to an already existing NAF SA. The check can be done by comparing the received (RAND, AUTN(*), Appl_Lbl) triplet with the corresponding triplets associated with existing NAF SAs.

4. The UE reads the GPI version number and selects the corresponding GPI MAC and ciphering algorithms. If the UE does not support this GPI version, the GPI is silently dropped and the GPI processing is terminated.

5. If the UICC application is active or can be activated the UE initiates derivation of the Ks by issuing an Authenticate command to the UICC. The type of Authenticate command is determined by the indicated U/M-mode in the GPI, i.e. if GBA_ME or GBA_U should be used. If the authenticate command returns a failure the GPI processing ends.

   If U/M indicates use of GBA_U, the generated Ks will effectively be generated on the UICC and not deleted until next GBA_U Ks is established using Authenticate command. The ME shall restrict NAF-key generation procedures using the generated Ks on the UICC to only be allowed for the NAF SA generation associated with the ongoing GBA-Push procedure.

6. The ME initiates the derivation of the GPI protection keys and other parameters needed for GPI integrity checking and deciphering of the confidentiality protected parts. This processing is defined in clause 5.3

7. The ME checks the integrity of the GPI message. If the integrity check fails, the following procedure is followed:

   a. With GBA_ME, the derived Ks is stored and marked as incomplete and the GPI processing ends.

   b. With GBA_U, the Ks was stored by the authenticate command. The Ks identity, which normally would be B-TID (see TS 33.220 [1]) is set to RAND@'undefined'. The GPI processing ends.

8. The ME deciphers the confidentiality protected parts of the GPI using the algorithms defined by the GPI version number and the GPI confidentiality protection keys.

9. The UE initiates the derivation of the NAF-Key (s), Ks(_int/ext)_NAF, using the NAF_Id received in the GPI. The key derivation is defined as specified in TS 33.220 [1].

10. The NAF SA consisting of the NAF-key(s) and associated parameters is stored.

NOTE 2: When GBA_U is used, two NAF-keys will be generated i.e. a Ks_ext_NAF will be stored in the ME and a Ks_int_NAF will be stored on the UICC. Both keys will be part of the NAF SA.

5.2 Data objects

5.2.1 GBA Push Information (GPI)

The definition of GPI information is given in table 5.2.1.1 Note that GPI does not contain any user identity or transport address as these entities are not needed by the GBA processing in the UE. They are only relevant for the transport of the GPI.

Table 5.2.1.1: GPI information

Parameter name	Description	Notes
Ver	Version of GPI	The version number is introduced to allow changes of GPI format and protection algorithms.
RAND	RAND in UMTS AKA	Defined in TS 33.102 [6]
AUTN(*)	AUTN or AUTN*	Defined in TS 33.220 [1]
App_Lbl	Identifier for UICC application to use	This variable may be left empty if the UICC application to use is evident from context or agreement.
U/M	Indicator for use of GBA_ME or GBA_U	
NAF_Id	Concatenation of NAF FQDN and Ua security protocol Id	Defined in TS 33.220 [1]; **Confidentiality protected**
Key_LT	Requested NAF-Key life time	**Confidentiality protected**
P-TID	Indicates request for private user identity	To be used by UE when responding to NAF. The identifier is included only to enable that it is confidentiality protected in the GPI. See also clauses 5.2.2 and 5.2.4. **Confidentiality protected**
MAC	Message authentication code over GPI	The integrity protection covers the complete GPI

This specification only defines a single version of GPI, i.e. version 1. In version 1, the MAC field is 32 bits.

5.2.2 NAF SA identities

A NAF SA holds NAF-key(s) and can have unique identities for uplink and downlink references, this to support unlinkability between uplink and downlink protection measures.

NAF SA identifiers are:

RAND@'naf': Identifies NAF SA in the UE (used by NAF).

Value of P-TID: Identifies NAF SA in the NAF (used by UE).

NOTE: 'naf' indicates a string of the characters naf; P-TID is assigned by the NAF and should be unique within the NAF.

5.2.3 NAF SA

The NAF needs to keep some additional information in its NAF SA compared with the UE. The UE identity used in the BSF request for GPI must be stored to allow the NAF to determine from which UE a response is coming and also to link sequences of SA's for the same UE. The NAF also needs to store the transport address to which the GPI should be directed. If the NAF uses retransmission to achieve better delivery reliability, it also needs to store the encrypted version of the part of the GPI, which is confidentiality protected. It also has to store the GPI MAC.

Table 5.2.3-1: NAF SA definition

Parameter name	NAF	UE	Description	Notes
UE_ Id	m	o	The user identity used in NAF request.	
UE_Priv_Id	o	-	Private user identity (IMSI/IMPI) for used UE_Id	
UE_Trp	m	-	Transport address to which GPI should be delivered	The transport address used by the NAF when pushing GPI to the UE
RAND	m	m	RAND in UMTS AKA	From GPI
AUTN(*)	m	m	AUTN or AUTN*	From GPI
Appl_Lbl	m	m	UICC application identifier	From GPI or other implicit agreement or information.
NAF_Id	m	m	Concatenation of NAF FQDN and Ua security protocol Id	
Enc_GPI	m	-	Encrypted part of GPI plus MAC	
Mac_GPI	m	-	BSF generated MAC over GPI	
UL_SA_Id	m	m	Uplink NAF SA identity	
DL_SA_Id	m	m	Downlink NAF SA identity	
Ks_NAF / Ks_ext_NAF	m	m	External NAF-key	Ks_NAF is generated in GBA_ME based GBA-Push Ks_ext_NAF is generated in GBA_U based GBA_Push
Ks_int_NAF	o	o	UICC internal NAF-key	Ks_int_NAF is generated in GBA_U based GBA_Push
Key_LT	m	m	Requested NAF-Key life time	
Repl_Cnt	o	o	Replay counter for outbound messages: First outbound message gets value 1	Outbound; Defined when Ua protocol is GPL
Repl_Win	o	o	Replay window for incoming messages: Initially the window is empty with highest received replay counter =0	Inbound; Defined when Ua protocol is GPL

5.3 GPI Integrity and Confidentiality Protection

5.3.1 General considerations

Integrity and confidentiality protection of the GPI is between the BSF and the UE. The keying material used for the protection must not leave the BSF and the UE, which implies that the NAF in particular and all other parties different from the UE and the BSF will not be able to modify the GPI (due to the integrity protection) or read its confidentiality protected parts.

NOTE: Transferring the NAF_Id in the clear together with a long term user identity/transport address may give rise to a privacy problem in a broadcast network or in an access network that does not apply confidentiality protection

5.3.2 Key material generation

The key material for confidentiality and integrity protection of GPI is derived from the Ks, which the GPI defines. The key derivations in version 1 of the GPI use the KDF defined in Annex B3 of TS 33.220 [1] with the below defined modifications of the NAF_ID (variable P3). The used NAF_ID as defined below shall be UTF- 8 encoded. All keys are 128 bits. The 128 least significant bits of the KDF output are used as key bits. The following keys are defined:

GPI_INT_Key: The NAF_ID shall equal 'GPI_integrity'.

GPI_ENC_Key: The NAF_ID shall equal 'GPI_confidentiality'

GPI_IV: The NAF_ID shall equal 'GPI_IV'

NOTE: It is appropriate to generate the IV this way as the keys will only be used to protect a single message.

5.3.3 GPI Integrity protection

GPI integrity protection is mandatory. The integrity protection is calculated after GPI has been confidentiality protected as defined in clause 5.3.4.

The GPI integrity protection in version 1 of the GPI uses algorithm HMAC-SHA256-32 with a 128-bit key as defined in [7], [8] and [9]. The MAC is computed over the complete GPI as defined in clause 5.2.1, During the computation of the MAC, the MAC field shall be treated as containing all zeros.

5.3.4 GPI Confidentiality protection

GPI confidentiality protection is mandatory.

The confidentiality protection shall be applied onGPI elements as indicated in table 5.2.1.1.

The GPI confidentiality algorithm in version 1 of the GPI is CTR-AES128 [10], [11]. The key to be used is GPI_ENC_Key and the start value T_1 for the counter is GPI_IV. The standard incrementing function is used with m=16, according to appendix B in [10], i.e. the 16 least significant bits in T behave like a counter while the 112 most significant bits are static and equal the 112 most significant bits of the GPI_IV.

Editor's NOTE: The details of the ciphering process needs to be defined. The processing and the used algorithms should be reviewed by SAGE.

5.3.5 GPI message format and coding

The GPI message is laid out as shown in Figure 5.3.5-1. Each field is encoded in network byte order (i.e., big endian) and with the most significant bit being bit number zero. All fields are octet aligned. The fields of the message are the following.

Ver (4 bits): The version of the GPI message encoded as a 4 bit binary number. The version of any message conforming to this specification shall use the value 1, i.e., the first nibble of the message is 0x1.

Reserved (3 bits): These bits are reserved for future versions of this specification. Implementations conforming to this specification shall set these bits to zero before transmitting a message, and the receiver of the message shall ignore these bits.

U/M (1 bit): 0 = GBA_ME, 1 = GBA_U

RAND: 16 octets.

AUTN: 8 octets.

Length App_Lbl: 1 octet containing length of App_Lbl in number of octets.

App_Lbl (variable length): UTF-8 encoded character string.

Length NAF_Id: 1 octet containing length of NAF_Id in number of octets.

NAF_Id (variable length): UTF-8 encoded FQDN of NAF concatenated with the 5 octets of the Ua security protocol identifier.

Key_LT: 4 octets. Key expiry time expressed in number of minutes since January 1, 1900, following the Network Time Protocol (NTP) [12].

NOTE: January 1, 1900 is the staring point for counting time in NTP

Length PTID: 1 octet containing length of PTID in number of octets.

PTID (variable length): UTF-8 encoded character strings

MAC: 4 octets.

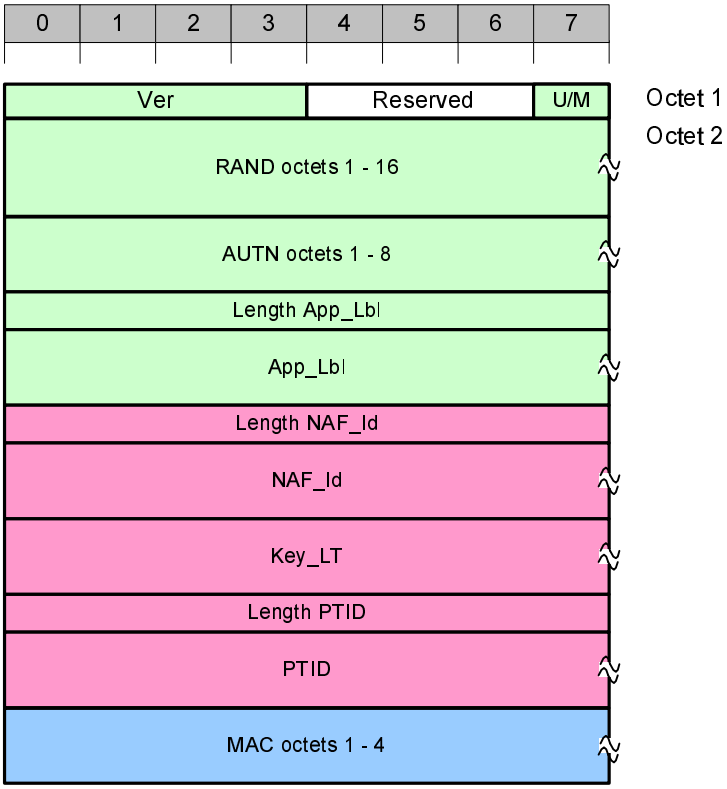

Figure 5.3.5-1. GPI message layout

5.4 Procedures using the NAF SA

The established NAF SA can be used by the terminal to set up communication over Ua.

If the terminal want to initiate a Ua connection based on a NAF SA established via TS 33.223 then the procedures for the terminal shall follow the principles defined in clause 4.5.3 in TS 33.220 [1] and clauses 5.3 and 5.4 in TS 33.222 [13] with the following change:

- Instead of referencing the SA (NAF-Key) to be used by a B-TID as described in TS 33.220, clause 4.5.3,.the UE uses P-TID. P-TID was sent in the GPI coming from NAF and it will uniquely identify the SA and the user identity to the NAF

- Instead of supplying the B-TID as user name as described in TS 33.222, clause 5.3, the UE uses P-TID. The NAF will then know the user identity and can retrieve the key from the NAF SA.

- Instead of using the B-TID as PSK identity as described in TS 33.222 clause 5.4 the UE uses P-TID. The NAF will then know the user identity and can retrieve the key from the NAF SA.

Annex A (informative): Rationale behind choice of the Disposable-Ks model

GBA-Push utilizes the Disposable-Ks model in which a Ks is only used once to derive a single set of NAF-keys. This means that after a NAF-key derivation, the used Ks is erased or its further usage is blocked. Furthermore, a Single Ks model is adopted for GBA_U based GBA-Push. This means that only a single GBA_U Ks can exist at a given time. The rationale behind the Single-Ks model for GBA_U based GBA-Push is to make it possible to reuse Rel-6 UICC's supporting GBA_U for GBA_U based GBA-Push.

For GBA_ME based GBA-Push the specification assumes that the ME can perform the necessary operations without having to erase a Ks generated by a normal GBA_ME bootstrapping.

The rationale behind the adoption of the Disposable-Ks model is to avoid synchronization problems as the GBA-Push may be over unreliable channels with non-delivery or undefined delay in delivery time, which may render the UE and the BSF unsynchronized with respect to the existence of a single valid GBA-Push Ks. Another situation when the UE and the BSF may become unsynchronized is when the BSF performs a normal bootstrapping and a NAF initiates a GBA-Push more or less simultaneously with the NAF requesting GPI before the UE performs the bootstrap and the GBA-Push message is delivered after the normal bootstrap. The Disposable-Ks model solves most of these out-of synch problems.

One situation when an out-of-synch problem will appear even with the adoption of the Disposable-Ks model is when the BSF may erase a valid Ks while the UE keeps it due to that the GBA-Push message can not be validated at the UE. This will lead to an error situation if the UE tries to use such a Ks. However, the error situation will easily be corrected as the NAF will get an error message from the BSF telling that the Ks (indicated by B-TID) is not available. The NAF would then return this error message and the terminal would perform a new bootstrap.

Alternatives to the chosen key handling model discussed were all based on allowing one or more GBA-Push generated Ks's and thus keeping a set of security contexts in the UE or on the UICC. Keeping one or more GBA-Push generated Ks's may make the out-of-synch problem go away or at least become much smaller. The downside is of course that as GBA_U based GBA-Push is essential from a security point of view, adoption of those models would have required new functionality on the UICC which was deemed making the introduction and adoption of GBA-Push more difficult. When also taking the minor functional drawbacks of the chosen key handling model into account the extra cost and complexity introduced by the other models were not judged to be a sufficient motivation to introduce new UICCs.

Annex B (informative): GBA-Push UE registration procedure

To be able to use GBA Push based services the user and the service provider need to share information. This is done in a registration procedure. The registration procedure could be explicit and involve the user or it could be automatic relying on user information provided by the user's operator. If the registration is initiated by the operator, the operator will have access to all needed registration information. When a user registers with a public identity this might not be the case, especially if the NAF is a third party service provider. One way of alleviating the problem would be to have users perform the registration over an authenticated/secured connection established with normal GBA. Then the BSF could provide the NAF with all needed information. Note however that this functionality is not standardized and that all needed information might not be available over the currently standardized interfaces.

At the registration the Push NAF needs to record the user identity (UE_Id), a push delivery method and the associated transport address (UE_Trp). The user identity may be either a public identity or a private identity.

A public IMS user identity (IMPU) can only be used if it maps to a unique private identity (IMPI). This MUST be checked in the registration procedure as the service will fail if the condition is not fulfilled.

When the UE identity is ann MSISDN this public identity will map uniquely to an IMSI but with number portability, being active information about the associated operator will be needed in any case to identify to which BSF the user belongs (i.e. to which operator the user has a contract). Knowing the operator will enable the NAF to derive the FQDN of the BSF in the operators network.

If the UE to be registered is equipped with a UICC holding more than one UICC application capable of running AKA the registration process should, if the UICC application to use if this is not uniquely determined by the UE transport method and/or UE_Id, determine which UICC application to use and how the NAF contacts the corresponding the BSF (needs to know the FQDN of the BSF). If explicit signalling is needed to identify the used USIM / ISIM, the App_Lbl to use should be agreed and recorded.

Annex Z (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
2006-05					Creation of document on the basis of SA3#43 discussion	0.0.0	0.0.1
2006-08					Integration of S3-060498, addition of editors' notes (SA3#44 meeting) and editorials.	0.0.1	0.1.0
2006-11					Integration of S3-060630, S3-060631, S3-060634, and S3-060676.	0.1.0	0.2.0
2007-02					Integration of S3-070041, S3-070049, S3-070051 and S3-070068 and their modifications as discussed in the meeting	0.2.0	0.3.0
2007-05					Integration of S3-070332, S3-070360, S3-070361 and S3-070372 and their modifications as discussed in the SA3#47 meeting	0.3.0	0.4.0
2007-07					Integration of S3-070510, S3-070538, S3-070557, S3-070652 and S3-070563 and their modifications as discussed in SA3#48 meeting	0.4.0	0.5.0
2007-10					Integration of S3-070710, S3-070711, S3-070773 and related decisions, presentation of TS 33.223 to SA Plenary for information	0.5.0	0.6.0
2007-11	38	SP-070834			Clean-up from MCC for presentation for information to TSG#38	0.6.0	1.0.0
2008-02					This version is against the proposed split of TS 33.223 v1.0.0. as proposed in S3-080070. Integration of S3-080051 and S3-080117 and their modifications as discussed in SA3#50 meeting	1.0.0	1.1.0
2008-03					Integration of S3-080133 and comments from email review	1.1.0	1.20
2008-05					Integration of GBA-Push principles agreed at S3#51 and relevant parts of pCR's S3-080304, S3-080305 and S3-080385	1.2.0	1.3.0
2008-05					SA3 agreement and clean up for presentation to SA#40	1.3.0	2.0.0
2008-06	SP-40	SP-080259			Approval at SA#40	2.0.0	8.0.0
2008-09	SP-41	SP-080483	0004	1	CR 33.223: UE registration at Push NAF	8.0.0	8.1.0
2008-09	SP-41	SP-080483	0002	1	CR 33.223: GPI Protection	8.0.0	8.1.0
2008-12	SP-42	SP-080741	0005	1	GBA-Push resolution of editors notes and corrections	8.1.0	8.2.0
2008-12	SP-42	SP-080741	0006	-	Introduction of UE-Id type indicator	8.1.0	8.2.0
2008-12	SP-42	SP-080741	0007	-	Push NAF authorization	8.1.0	8.2.0

History

Document history		
V8.2.0	January 2009	Publication