3GPP TS ab.cde V0.1.0 (2003-10)

Technical Specification

3rd Generation Partnership Project;
Technical Specification Group Services and System Aspects;
Generic Authentication Architecture (GAA);
Generic Bootstrapping Architecture
(Release 6)

Remove GSM logo from the cover page for pure 3rd Generation documents.

Select keywords from list provided in specs database.

Keywords

<keyword[, keyword]>

3GPP

Postal address

3GPP support office address

650 Route des Lucioles - Sophia Antipolis
Valbonne - FRANCE
Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Internet

http://www.3gpp.org

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

This clause is optional. If it exists, it is always the second unnumbered clause.

1 Scope

This clause shall start on a new page.

The present document …

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.*

[<seq>] <doctype> <#>[([up to and including]{yyyy[-mm]|V<a[.b[.c]]>}[onwards])]: "<Title>".

[1] 3GPP TR 41.001: "GSM Release specifications".

[2] 3GPP TR 21.912 (V3.1.0): "Example 2, using fixed text".

[3] 3GPP TS 31.102: "Characteristics of the USIM Application".

[4] 3GPP TS 33.102: "Security Architecture".

[PKCS10] "PKCS#10 v1.7: Certification Request Syntax Standard", RSA Laboratories, May 2000.

[RFC2510] Adams C., Farrell S., "Internet X.509 Public Key Infrastructure Certificate Management Protocols", RFC 2510, March 1999.

[RFC2511] Myers M., et al., "Internet X.509 Certificate Request Message Format", RFC 2511, March 1999.

[RFC2527] Chokhani S., et al, "Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework", RFC 2527, March 1999.

[RFC2617] Franks J., et al, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.

[RFC3280] Housley R., et al, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002.

[RFC 3310] A. Niemi, et al, "Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA)", RFC3310, September 2002.

[WAPCert] WAP-211-WAPCert, 22.5.2001: http://www1.wapforum.org/tech/terms.asp?doc=WAP-211-WAPCert-20010522-a.pdf

[WIM] WAP-260-WIM-20010712, 12.7.2001: http://www1.wapforum.org/tech/documents/WAP-260-WIM-20010712-a.pdf

[WPKI] WAP-217-WPKI, 24.4.2001: http://www1.wapforum.org/tech/documents/WAP-217-WPKI-20010424-a.pdf

[X.509] ITU-T Recommendation X.509 (1997) | ISO/IEC 9594-8:1997, "Information Technology - Open Systems Interconnection - The Directory: Authentication Framework", 1997.

[TR GAA]

[TS SSC]

[TS HTTPS]

3 Definitions, symbols and abbreviations

Delete from the above heading those words which are not applicable.

Subclause numbering depends on applicability and should be renumbered accordingly.

3.1 Definitions

For the purposes of the present document, the [following] terms and definitions [given in ... and the following] apply.

Definition format

***<defined term>**: <definition>.*

example: text used to clarify abstract rules by applying them literally.

3.2 Symbols

For the purposes of the present document, the following symbols apply:

Symbol format

<symbol>	<Explanation>

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

Abbreviation format

<ACRONYM>	<Explanation>
AK	Anonymity Key
AKA	Authentication and Key Agreement
BSF	Bootstrapping server functionality BSF is hosted in a network element under the control of an MNO.
BSP	BootStrapping Procedure
CA	Certificate Authority
CMP	Certificate Management Protocols
HSS	Home Subscriber System
IK	Integrity Key
MNO	Mobile network operator
NAF	Operator-controlled network application function functionality. NAF is hosted in a network element under the control of an MNO.
PKCS	Public-Key Cryptography Standards
PKI	Public Key Infrastructure
SCP	Subscriber Certificate Procedure
UE	User Equipment¨

4 Generic ~~AKA b~~Bootstrapping ~~functions~~Architecture

The 3GPP authentication infrastructure, including the 3GPP Authentication Centre (AuC), the USIM and the 3GPP AKA protocol run between them, is a very valuable asset of 3GPP operators. It has been recognised that this infrastructure could be leveraged to enable application functions in the network and on the user side to communicate in situations where they would not be able to do so without the support of the 3GPP authentication infrastructure. Therefore, 3GPP can provide the "bootstrapping of application security" to authenticate the subscriber by defining a generic bootstrapping function based on AKA protocol.

4.1 Requirements and principles for bootstrapping

Editor's note: The description of AKA bootstrapping shall be added here.

- The bootstrapping function shall not depend on the particular network application function
- The server implementing the bootstrapping function needs to be trusted by the home operator to handle authentication vectors.
- The server implementing the network application function needs only to be trusted by the home operator to handle derived key material.
- It shall be possible to support network application functions in the operator's home network
- The architecture shall not preclude the support of network application function in the visited network, or possibly even in a third network.
- To the extent possible, existing protocols and infrastructure should be reused.
- In order to ensure wide applicability, all involved protocols are preferred to run over IP.

4.1.1 Access Independence

Bootstrapping procedure is access independent. Bootstrapping procedure requires IP connectivity from UE.

4.1.2 Authentication methods

Authentication method that is used to authenticate the bootstrapping function must be dependent on cellular subscription. In other words, authentication to bootstrapping function shall not be possible without valid cellular subscription. Authentication shall be based on AKA protocol.

4.1.3 Roaming

The roaming subscriber shall be able to utilize the bootstrapping function in home network.

Editor's note: For the first phase of standardisation, only the case is considered where bootstrapping server functionality and network application function are located in the same network as the HSS. In later phases, other configurations may be considered.

4.2 Bootstrapping architecture

4.2.1 Reference model

Figure 1 shows a simple network model of the entities involved in the bootstrapping approach, and the protocols used among them.

~~Editor's note: The names for the reference points, A, B, C, and D need to be decided.~~

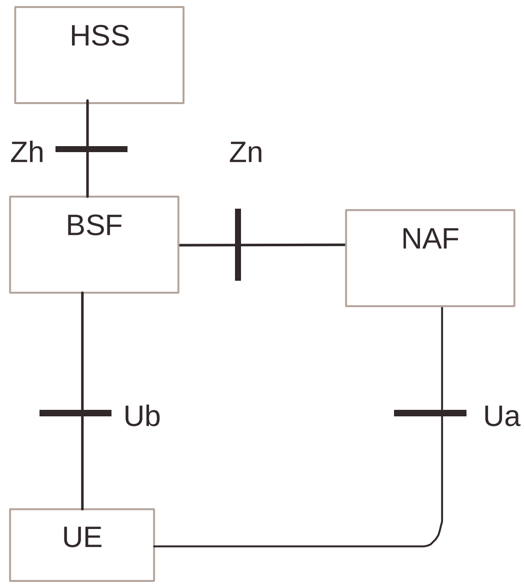

Figure 1: Simple network model for bootstrapping

Figure 2 illustrates a protocol stacks structure in network elements that are involved in bootstrapping of application security from 3G AKA and support for subscriber certificates.

Editor's note: The current protocol stack figure is placed here as a holder. The actual protocols will be defined later.

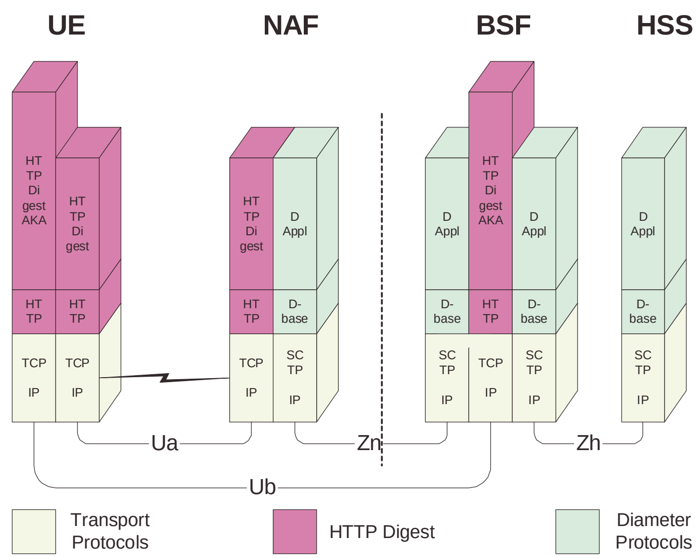

Figure 2: Protocol stack architecture

4.2.2 Network elements

4.2.2.1 Bootstrapping server function (BSF)

A generic bootstrapping server function (BSF) and the UE shall mutually authenticate using the AKA protocol, and agree on session keys that are afterwards applied between UE and an operator-controlled network application function (NAF). The key material must be generated specifically for each NAF independently.

Editor's note: key generation for NAF is ffs. Potential solutions may include:

- Separate run of ~~protocol A~~Ub interface for each request of key material from a NAF
- Derivation of NAF-specific keys in BSF

4.2.2.2 Network application function (NAF)

After the bootstrapping has been completed, the UE and an operator-controlled network application function (NAF) can run some application specific protocol where the authentication of messages will be based on those session keys generated during the mutual authentication between UE and BSF.

General assumptions for the functionality of an operator-controlled network application function (NAF):

- There is no previous security association between the UE and the NAF.
- NAF shall able to locate and communicate securely with subscriber's BSF.
- NAF shall be able to acquire a shared key material established between UE and the bootstrapping server function (BSF) during running application-specific protocol.

4.2.2.3 HSS

HSS shall store new parameters in subscriber profile related to the usage of bootstrapping function. Possibly also parameters related to the usage of some network application function are stored in HSS.

Editor's note: Needed new parameters are FFS.

4.2.2.4 UE

The required new functionalities from UE are:

- The support of HTTP Digest AKA protocol,
- The capability to derive new key material to be used with ~~protocol B~~Ua interface from CK and IK, and
- Support of NAF specific application protocol (see ~~annex A~~[TS SSC]).

4.2.3 Reference points

4.2.3.1 ~~A~~Ub interface

The reference point ~~A~~Ub is between the UE and the BSF. The functionality is radio access independent and can be run in both CS and PS domains.

Editor's notes: The solution for CS domain is ffs.

4.2.3.1.1 Functionality

Reference point ~~A~~Ub provides mutual authentication between the UE and the BSF entities. It allows the UE to bootstrap the session keys based on the 3G infrastructure. The session key as result of key agreement functionality, is used to support further applications e.g. certificate issuer.

4.2.3.1.2 Protocol

~~Protocol A~~Ub interface is in format of HTTP Digest AKA, which is specified in [RFC3310]. It is based on the 3GPP AKA [4] protocol that requires information from USIM and/or ISIM. The interface to the USIM is as specified for 3G [3].

4.2.3.2 ~~B~~Ua interface

~~Protocol B~~Ua interface is the application protocol which is secured using the keys material agreed between UE and BSF as a result of the run of ~~protocol A~~Ub interface. For instance, in the case of support for subscriber certificates [TS SSC], it is a protocol, which allows the user to request certificates from the NAF. In this case NAF would be the PKI portal.

4.2.3.3 ~~C~~Zh interface

~~Protocol C~~Zh interface is used between the BSF and the HSS to allow the BSF to fetch the required authentication information and subscriber profile information from the HSS. The interface to the 3G Authentication Centre is HSS-internal, and it need not be standardised as part of this architecture.

4.2.3.4 ~~D~~Zn interface

~~Protocol D~~Zn interface is used by the NAF to fetch the key material agreed in ~~protocol A~~Ub interface from the BSF. It may also be used to fetch subscriber profile information from BSF.

4.3 Procedures

This chapter specifies in detail the format of the bootstrapping procedure that is further utilized by various applications. It contains the AKA authentication procedure with BSF, and latter the key material generation procedure.

4.3.1 Bootstrapping procedures

When a UE wants to interact with an NAF, it shall first perform a bootstrapping authentication (see Figure 3):Editor's notes: ~~Protocol C~~Zh interface related procedure will be added here in future development. It may re-use Cx interface that is specified in TS 29.228. It shall also perform a bootstrapping authentication when it has received a key update indication from the NAF (cf. section 4.3.2).

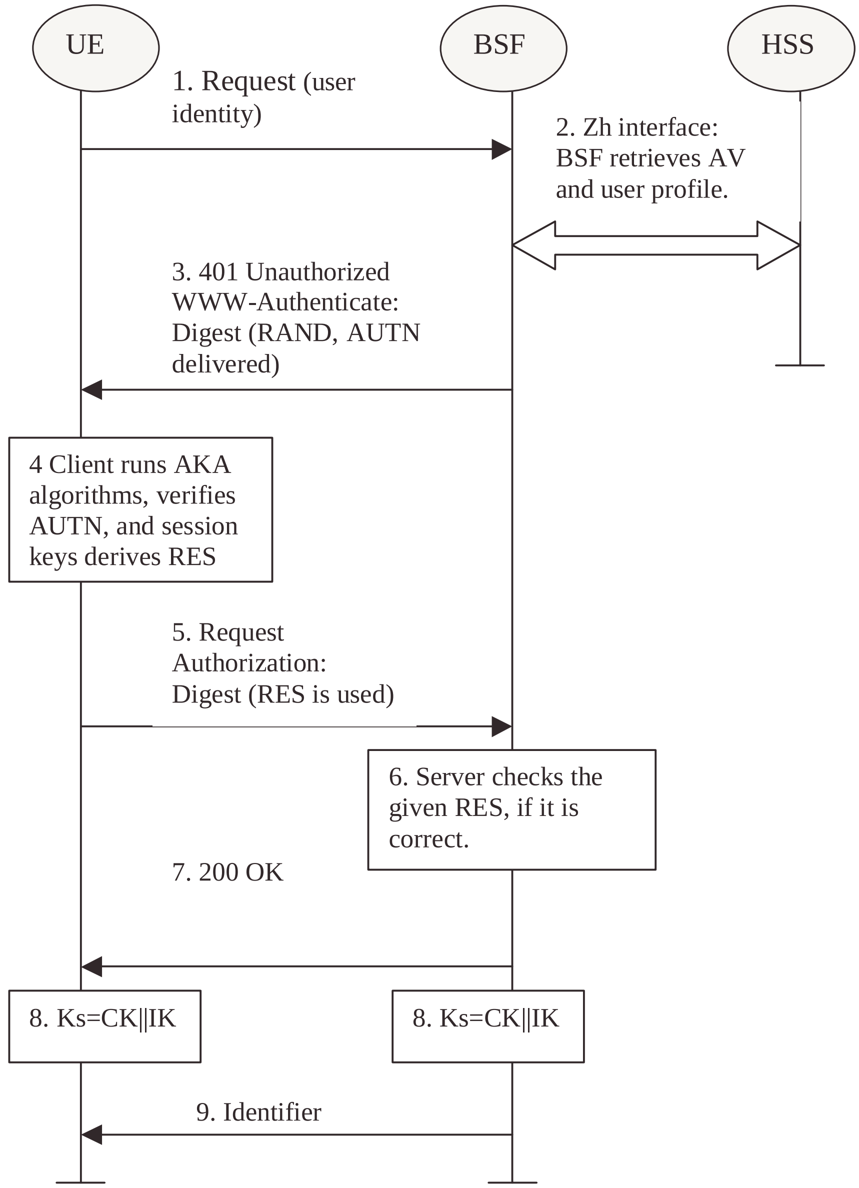

Figure 3: The bootsrapping procedure

1. The UE sends an HTTP request towards the BSF.

2. BSF retrieves the user profile and a challenge, i.e. the Authentication Vector (AV, AV = RAND||AUTN||XRES||CK||IK) by ~~protocol C~~Zh interface from the HSS.

3. Then BSF forwards the RAND and AUTN to the UE in the 401 message (without the CK, IK and XRES). This is to demand the UE to authenticate itself.

4. The UE calculates the message authentication code (MAC) so as to verify the challenge from authenticated network; the UE also calculates CK, IK and RES. This will result in session keys IK and CK in both BSF and UE.

5. The UE sends request again, with the Digest AKA RES as the response to the BSF.

6. If the RES equals to the XRES that is in the AV, the UE is authenticated.

7. The BSF shall send 200 OK message to the UE to indicate the success of the authentication.

8. The key material Ks is generated in both BSF and UE by concatenating CK and IK. The Ks is used for securing the ~~protocol B~~Ua interface.

Editor's note: The key material Ks is 256 bits long. It is up each NAF to make the usage of the key material specifically.

9. BSF may supply a transaction identifier to UE in the cause of ~~protocol A~~Ub interface.

4.3.2 Procedures using bootstrapped Security Association

After UE is authenticated with the BSF, every time the UE wants to interact with an NAF the following steps are executed as depicted in Figure 4

UE starts ~~protocol B~~Ua interface with the NAF

- In general, UE and NAF will not yet share the key(s) required to protect ~~protocol B~~Ua interface. If they already do, there is no need for NAF to invoke ~~protocol D~~Zn interface.
- If the NAF shares a key with the UE, but an update of that key it sends a suitable key update request to the UE and terminates ~~protocol B~~Ua interface. The form of this indication may depend on the particular ~~protocol B~~Ua interface and is ffs.
- It is assumed that UE supplies sufficient information to NAF, e.g. a transaction identifier, to allow the NAF to retrieve specific key material from BSF.
- The UE derives the keys required to protect ~~protocol B~~Ua interface from the key material.

NAF starts ~~protocol D~~Zn interface with BSF

- The NAF requests key material corresponding to the information supplied by the UE to the NAF (e.g. a transaction identifier) in the start of ~~protocol B~~Ua interface.
- The BSF supplies to NAF the requested key material. If the key identified by the transaction identifier supplied by the NAF is not available at the BSF, the BSF shall indicate this in the reply to the NAF. The NAF then indicates a key update request to the UE.
- The NAF derives the keys required to protect ~~protocol B~~Ua interface from the key material in the same way as the UE did.

NAF continues ~~protocol B~~Ua interface with UE

Once the run of ~~protocol B~~Ua interface is completed the purpose of bootstrapping is fulfilled as it enabled UE and NAF to run ~~protocol B~~Ua interface in a secure way.

Editor's note: Message sequence diagram presentation and its details will be finalized later.

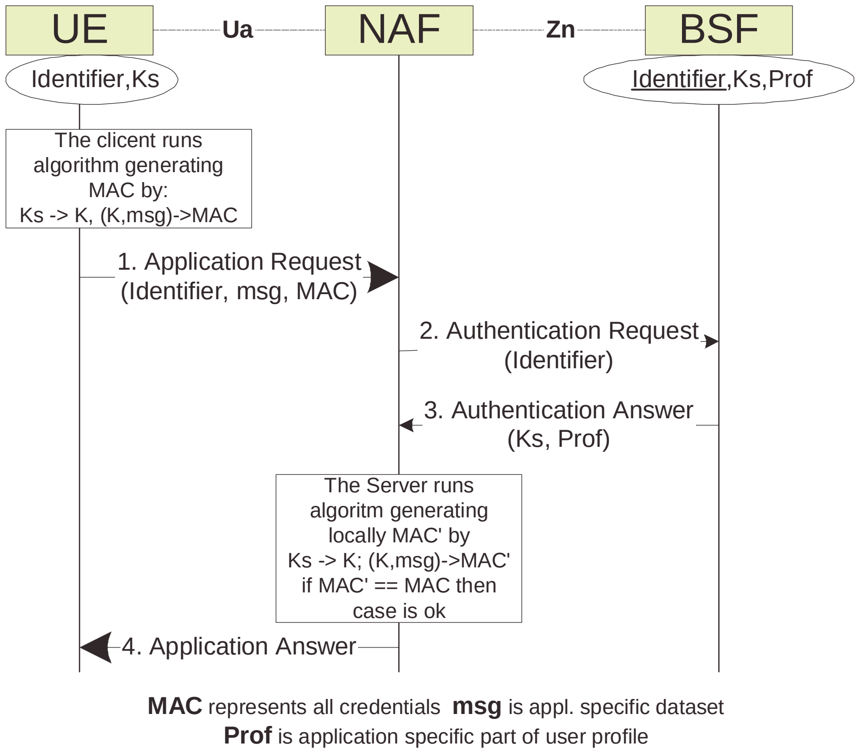

Figure 4: The bootstrapping usage procedure

Annex <~~B~~A> (informative): Generic secure message exchange using HTTP Digest Authentication

~~B~~A.1 Introduction

Editor's note: This annex describes how HTTP Digest Authentication can be used between UE and any NAF whose ~~protocol B~~Ua interface is based on HTTP messaging. ~~Protocol B~~Ua interface may depend upon the final choice of scheme made by SA WG3 and this will need to be reviewed later by SA WG3.

HTTP Digest Authentication model can also be used as a generic authentication and integrity protection method towards any new NAF. If a new NAF uses BSF-based security association, it could use this generic method to authenticate the UE (and UE authenticate the NAF) and integrity protect any payload being transferred between NAF and UE. As a generic method, it will speed up the specification of new NAFs since the authentication and message integrity protection part of ~~protocol B~~Ua interface are taken care of by HTTP Digest Authentication. It will also ease the implementation of BSF-based authentication in NAFs because there would be one well-defined way to do it.

~~B~~A.2 Generic ~~protocol B~~Ua interface description

The sequence diagram in Figure 5 describes the generic secure message exchange with HTTP Digest Authentication. The conversation may take place inside a server-authenticated TLS [RFC2246] tunnel in which case TLS handshake has taken place before step 1.

In step 1, UE sends an empty HTTP request to a NAF. In step 2, NAF responds with HTTP response code 401 "Unauthorized" which contains a WWW-Authenticate header. The header instructs the UE to use HTTP Digest Authentication with a bootstrapped security association. Quality of protection (qop) attribute is set to "auth-int" meaning that the payload of the following HTTP requests and responses should integrity protected. The realm attribute contains two parts. The first part is a constant string "3GPP-bootstrapping" instructing the UE to use a bootstrapped security association. The second part is the DNS name of the NAF.

In step 3, the UE shall verify that the second part of the realm attribute does in fact correspond to the server it is talking to. In particular, if the conversation is taking place inside a server-authenticated TLS tunnel, the UE shall verify that the server name in the server's TLS certificate matches the server name in the realm attribute of the WWW-Authenticate header. The UE generates client-payload containing the message it wants to send to the server. Then it will generate the HTTP request by calculating the Authorization header values using the transaction identifier (base64 encoded) it received from the BSF as username and the session key K (base64 encoded) as the password, and send the request to NAF in step 4.

When NAF receives the request in step 5, it will verify the Authorization header by fetching the session key K from the bootstrapping server using ~~protocol D~~Zn interface and the transaction identifier. After successful retrieval, NAF calculates the corresponding digest values using K, and compares the calculated values with the received values in the Authorization header. The NAF shall also verify that the DNS name in the realm attribute matches its own. If the conversation is taking place inside a server-authenticated TLS tunnel, the NAF shall also verify that this DNS name is the same as that of the TLS server. If the verification succeeds, the incoming client-payload request is taken in for further processing. Thereafter, the NAF will generate a HTTP response containing the server-payload it wants to send back to the client in step 6. The NAF may use session key K to integrity protect and authenticate the response.

In step 7, UE receives the response and verifies the Authentication-Info header. If the verification succeeds, the UE can accept the server-payload for further processing.

Additional messages can be exchanged using steps 3 through 7 as many times as is necessary. The following HTTP request and responses must be constructed according to RFC 2617 (e.g., nc parameter must be incremented by one with each new HTTP request made by UE).

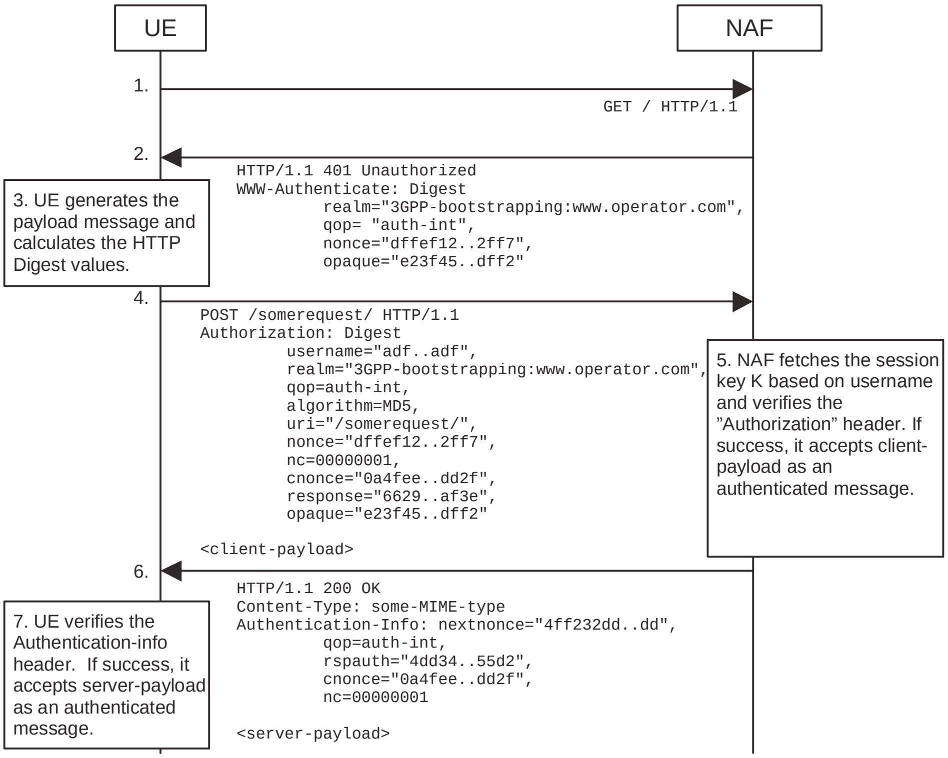

Figure 5: Generic secure message exchange using HTTP Digest Authentication and bootstrapped security association

Annex <X> (informative): Change history

Change history							
Date	**TSG #**	**TSG Doc.**	**CR**	**Rev**	**Subject/Comment**	**Old**	**New**
2003-10	*SA3#30*				*New draft TS: Generic Bootstrapping Architecture (GBA). Extracted from 33.109 sections 4 and Annex B.*		*0.1.0*
2003-10	*SA3#30*	*S3-030537*			*New interface names.*		*0.1.0*