

INTERNATIONAL TELECOMMUNICATION UNION

**TELECOMMUNICATION  
STANDARDIZATION SECTOR**

STUDY PERIOD 2009-2012

**NGN-GSI**

**TD 271 (NGN-GSI)**

**English only**

**Original: English**

---

**Question(s):** 16/13

Geneva, 18-29 January 2010

**TEMPORARY DOCUMENT**

**Source:** Editor

**Title:** Updated Draft Recommendation NGN Identity Management Mechanisms

---

---

<b>Contact:</b>	Takashi Egawa NEC Japan	Tel: +81 44 431 7770 Fax: +81 44 431 7771 Email: t-egawa@ct.jp.nec.com
-----------------	-------------------------------	--

---

<b>Contact:</b>	Zachary Zeltsan Alcatel-Lucent USA	Tel: + 1 908 582 2359 Email:zeltsan@alcatel-lucent.com
-----------------	--	---

<b>TSB Note:</b> All rights reserved. No part of this publication may be reproduced, by any means whatsoever, without the prior written permission of ITU.
--

1	Scope .....	7
2	References.....	7
3	Definitions .....	8
	3.1 Terms defined in other Recommendations .....	8
	3.2 Terms defined in this Recommendation .....	8
4	Abbreviations.....	8
5	Conventions .....	8
6	Mechanisms and Procedures supporting IdM Functions.....	9
6.1	Lifecycle Management .....	9
6.1.1	Enrolment .....	9
6.2	Authentication and Authentication Assurance .....	9
6.2.1	Authentication based on WS Security SAML Profile .....	9
6.2.1.1	SAML assertions .....	9
6.2.1.2	Subject confirmation methods of the SAML tokens .....	11
6.2.1.2.1	The holder-of-key subject confirmation method .....	12
6.2.1.2.2	The sender-vouches subject confirmation method .....	13
6.2.2	Certificate-based authentication .....	14
6.2.3	Password-based authentication.....	14
6.2.4	One-time Password.....	14
6.2.5	Authentication and Key Agreement (AKA).....	15
6.2.5.1	Overview of the AKA.....	15
6.2.5.2	Generation of the authentication vector by AuC/HLR.....	16
6.2.5.3	AKA operation in USIM .....	17
6.2.5.4	Sizes of the AKA cryptographic parameters .....	17
6.2.5.5	Universal Subscriber Identity Module (USIM).....	17
6.2.5.6	Use of the AKA in non-wireless environment .....	18
6.2.6	Integration of PKI-based authentication with IMS.....	18

6.2.6.1 Conventions .....	18
6.2.6.2 Entities involved in authentication .....	19
6.2.6.3 Establishing agreement on the CK and IK keys with the use of a shared secret between the End-User Function and S-5 (option 1) .....	19
6.2.6.4 Establishing agreement on the CK and IK keys without the use of a shared secret between the End-User Function and S-5 (option 2) .....	20
6.2.6.5 Comparison of the option 1 and option 2 .....	22
6.2.6.6 Requirements to the End-User Function .....	23
6.2.6.7 Requirements to the S-1 .....	23
6.2.6.8 Requirements on the SIP interfaces between the participating entities.....	24
6.2.6.9 Requirements on the Diameter interfaces between the participating entities.....	24
6.2.7 Integration of the PKI-based authentication and the SAML assertion mechanisms ....	24
Entities involved in the authentication and the information flow .....	25
Conventions .....	25
Mechanism's parameters.....	25
Figure 1 - The basic steps of data exchange for the PKI-based authentication with SAML- assertion .....	27
Additional requirements for the entities participating in the authentication.....	28
Additional requirements for the interfaces between the participating entities.....	29
6.2.8 Integration of <i>OpenID</i> -based authentication with IMS .....	29
6.2.8.1 Entities involved in the authentication and the information flow.....	29
6.2.8.2 Additional requirements for the entities participating in the authentication .....	31
6.2.8.3 Additional requirements for the interfaces between the participating entities .....	32
6.2.9 GBA.....	33
6.3 Correlation and Binding .....	35
6.4 Discovery.....	35
6.4.1 Intra-network Discovery.....	35
6.4.2 Inter-network Discovery.....	35
6.5 Policy Enforcement .....	35

6.6	IdM Communications and Information Exchange .....	36
6.6.1	External Interfaces .....	36
6.6.2	Internal Interfaces .....	36
<b>6.6.3</b>	<b>Security of IdM Communications and Exchange</b> .....	36
<b>6.6.3.1</b>	<b>SAML 2.0 (ITU-T Recommendation X.1141 [4])</b> .....	36
6.6.3.2	Identity Web Services Framework (known as ID-WSF).....	36
[2]	Web Services Security X.509 Certificate Token Profile 1.1, OASIS .....	40
6.7	User and Subscriber Control.....	40
6.8	Protection of Personally Identifiable Information (PII).....	40
6.9	Federated Identity Functions .....	40
6.9.1	Bridging and Interworking .....	41
6.9.2	Discovery of IdPs in Federated Environment.....	41
6.10	Identity Information Access Control .....	41
6.10.1	SAML-based mechanism for attribute sharing.....	41
6.10.2	Pseudonym management .....	41
6.10.3	Integrity of the SAML assertions .....	41
6.11	Single Sign-on .....	41
6.11.1	SAML-based mechanism .....	42
6.12	Single Sign-off.....	42
7	Security.....	42
	Appendix I: IdM Profiles for NGN.....	42
	Appendix II: Bibliography .....	42
	Appendix III.....	42
	Appendix IV: Example identities in NGN.....	43
<b>8</b>	<b>NGN Identities {It is proposed that the information in this section be consolidated, deleted or included in an Appendix}</b> .....	44
<b>8.1</b>	<b>Subscriber Identifiers</b> .....	44
<b>8.1.1</b>	<b>Public identifiers</b> .....	44
<b>8.1.2</b>	<b>Private identifiers</b> .....	44

<b>8.2</b>	<b>Network/Service Provider identifiers .....</b>	<b>45</b>
<b>8.2.1</b>	<b>Public network/service providers.....</b>	<b>45</b>
<b>8.2.2</b>	<b>Private/home network service providers.....</b>	<b>45</b>
<b>8.3</b>	<b>Object Identities.....</b>	<b>45</b>
<b>8.3.1</b>	<b>Terminal or Sensor Devices.....</b>	<b>46</b>
<b>8.3.2</b>	<b>Network-Based Equipment.....</b>	<b>46</b>
<b>8.3.3</b>	<b>Other Objects.....</b>	<b>46</b>
<b>Appendix V: X.509 v3 Message Authentication .....</b>		<b>46</b>

[This page intentionally left blank for this distribution version]

## NGN IDENTITY MANAGEMENT MECHANISMS

### 1 Scope

Draft Y.idmRequirements, describes NGN IdM requirements. **Editor's note:** the next sentence includes information of this deleted sentence.

This Recommendation describes the specific IdM mechanisms and suites of options that should be used to meet the requirements in Y.idmRequirements of NGN. In addition, it could provide best practices, guidelines to support interoperability and other needs.

**[EdNote:** Examples mechanisms and approaches recommended to or should be used to meet the requirements may include

- SAML
- X.509
- ID-WSF
- GBA, and
- E.115

]

### 2 References

The following ITU-T Recommendations and other references contain provisions, which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; all users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published.

[ATIS33102] ATIS.3GPP.33.102V710-2007, *Security Architecture*

[Y.2720] ITU-T Recommendation Y.2720 (2009), NGN Identity Management Framework.

[Y.idmRequirements] ITU-T Recommendation Y.idmRequirements, NGN Identity Management Requirements

[Y.2704] ITU-T Recommendation Y.2704 (2010), Security mechanisms and procedures for NGN

[Y.2702] ITU-T Recommendation Y.2702 (2008), Authentication and authorization requirements for NGN release 1

[Y.2012] Recommendation Y.2012, Functional Requirements and Architecture of the NGN of Release 1, 09/2006.

[ITU-T SAML] ITU-T Recommendation X.1141 (2006), *Security Assertion Markup Language (SAML 2.0)*

[ATIS33102] ATIS.3GPP.33.102V710-2007, *Security Architecture*

[X.509] ITU-T Recommendation X.509 (2005), Information Technology – Open Systems Interconnection – The Directory: Authentication Framework

[X.1141] ITU-T Recommendation X.1141 (2006), Security Assertion Markup Language (SAML 2.0)

[RFC 2616] IETF RFC 2616 (1999), *Hypertext Transfer Protocol -- HTTP/1.1*, <<http://tools.ietf.org/html/rfc2616>>

### 3 Definitions

#### 3.1 Terms defined in other Recommendations

#### 3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

### 4 Abbreviations

This Recommendation uses the following abbreviations and acronyms:

IdM Identity Management

IdP Identity Provider

SAML Security Assertion Markup Language

### 5 Conventions

In this document:

The keywords “**is required to**” indicate a requirement which must be strictly followed and from which no deviation is permitted if conformance to this document is to be claimed.

The keywords “**is recommended**” indicate a requirement which is recommended but which is not absolutely required. Thus this requirement need not be present to claim conformance.

The keywords “**is prohibited from**” indicate a requirement which must be strictly followed and from which no deviation is permitted if conformance to this document is to be claimed.

The keywords “**can optionally**” indicate an optional requirement which is permissible, without implying any sense of being recommended. This term is not intended to imply that the vendor’s implementation must provide the option and the feature can be optionally enabled by the network operator/service provider. Rather, it means the vendor may optionally provide the feature and still claim conformance with the specification.

In the body of this document and its annexes, the words *shall*, *shall not*, *should*, and *may* sometimes appear, in which case they are to be interpreted, respectively, as *is required to*, *is prohibited from*, *is recommended*, and *can optionally*. The appearance of such phrases or keywords in an appendix or in material explicitly marked as *informative* are to be interpreted as having no normative intent.

## 6 Mechanisms and Procedures supporting IdM Functions

### 6.1 Lifecycle Management

**Editor's note:** This section would identify and recommend procedures for identity and identity information lifecycle management (identity, identifiers, attributes, policy, etc)

#### 6.1.1 Enrolment

Enrolment is a process that provides means for an entity (e.g., user, device, object, process, etc.) to subscribe with an IdP. The enrolment is performed when an entity first time applies (subscribes) for a service. During the enrolment, an IdP must verify (subject to the policy) identity information provided by the entity. For example, if the entity is a user, such verification may require appearance of the user in the office where her or his authentication can be done using the legal documents (e.g., passport, driver license, etc.).

The enrolment also includes providing the entity with the means to prove its identity in the subsequent requests for a service. This can be done by assigning to the entity an IdP identifier and providing it with authentication credentials (e.g., password, UICC card, security token, etc.)

All relevant to the entity information submitted by the entity and generated by an IdP during the enrolment must be stored. Storing such information is also responsibility of the enrolment process.

The clause *Enrolment and proofing* of the ITU-T Recommendation Y.2720, *NGN Identity Management Framework* provides additional information on the enrolment process.

### 6.2 Authentication and Authentication Assurance

This clause describes mechanisms for authentication and assurance of identities and identity information. It references authentication mechanisms defined elsewhere.

IdP supports authentication methods such as authentication based on WS Security SAML Profile, Certificate-based authentication, or Password-based authentication (including OTP). The authentication method (or methods) are selected based on the assurance level requirements. The IdP may request assurance level information to find the authentication methods that satisfy the service provider's assurance level requirements.

**(Editor's note:** Future contributions on specific examples and capabilities in support of IdP querying the provider's assurance level information are requested.

Contributions on authentication assurance are requested.

In addition, contributions on the specific *negotiation protocols* and other mechanisms are requested.)

#### 6.2.1 Authentication based on WS Security SAML Profile

##### 6.2.1.1 SAML assertions

Security Assertion Markup Language (SAML) [ITU-T SAML] specifies format of assertions that can be used in Identity Management for exchanging security information. Among the IdM functions that can be implemented with the use of SAML are authentication, attribute sharing, and authorization, which correspond to three types of the statements about a subject of a SAML assertion:

- Authentication statement – conveys information that the assertion subject was authenticated by a particular means at a particular time.
- Attribute statement – conveys information that the assertion subject is associated with the listed attributes.
- Authorization decision statement – conveys information that access to a specified resource was granted to the assertion subject, or the subject was denied such access.

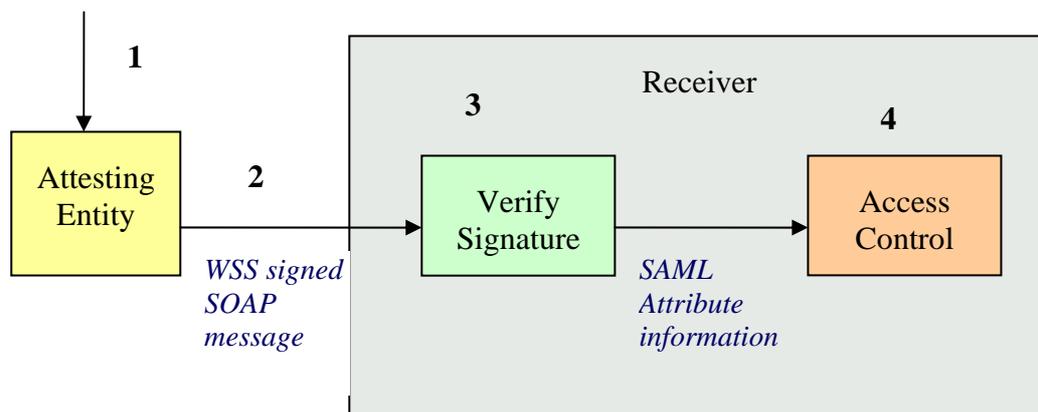
The content of a SAML assertion can be described at a high level as follows: assertion **A** was issued at time **t** by issuer **R** regarding subject **S** provided conditions **C** are valid.

The use of the SAML assertions for conveying authentication, attribute and authorization information in SOAP messages is a special and important application of SAML. When SOAP messages are exchanged over an unprotected transport, it is strongly recommended that XML signature [XML signature] is used to verify relationship between the SOAP message and the statements of the assertions carried in the message. The *Web Services Security (WSS): SAML Token Profile* [SAML token] standard describes how:

- SAML assertions (also referred to as SAML tokens) are carried in and referenced from a SOAP message.
- XML signature is used to bind a subject and the statements of a SAML assertion with a SOAP message.

A typical use of a SAML token with SOAP message constructed according to this specification is depicted by Figure 1 and described below.

In this example a signed SOAP message contains a SAML assertion with an attribute statement. Based on the information in this statement the receiver decides whether to allow access to the requested resource.



**Figure 1 - Typical steps of construction and processing of a SOAP message with a SAML token**

1. The Attesting Entity obtains a SAML assertion with an attribute statement and constructs and includes it in a SOAP message constructed according to [SAML token].
2. The Attesting Entity sends SOAP message to the Receiver.
3. The Receiver verifies digital signature.
4. The information of the SAML statement is used for access control.

### 6.2.1.2 Subject confirmation methods of the SAML tokens

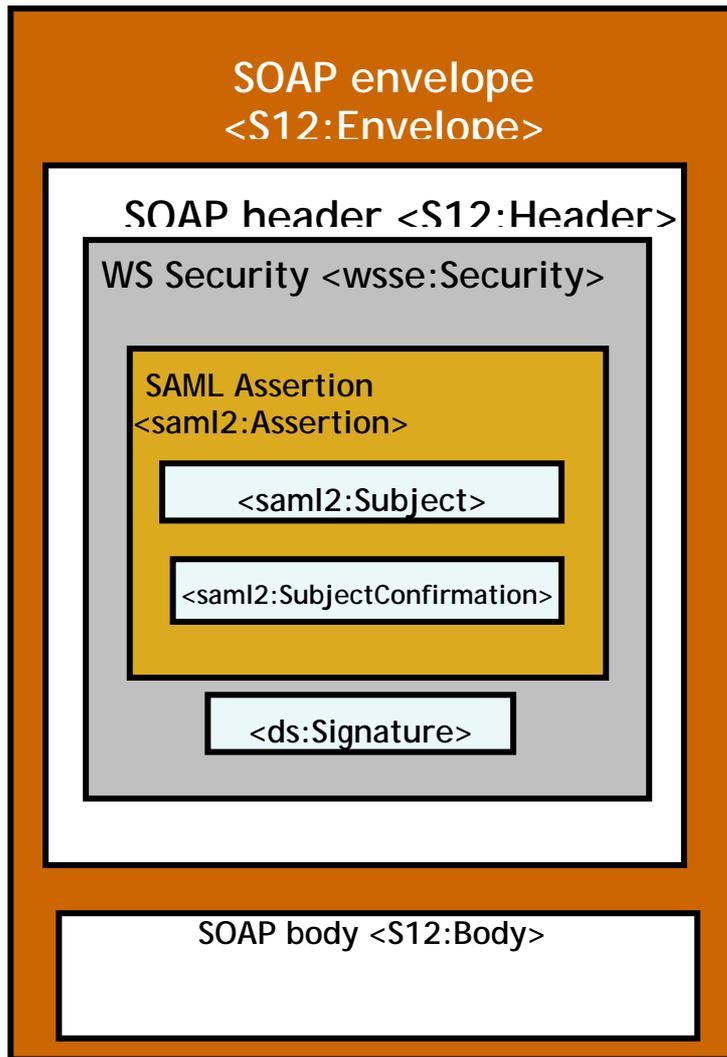
The OASIS Standard, *Web Services Security: SAML Token Profile 1.1* [SAML token] specifies how to attach a SAML [SAML] assertion to a SOAP message and defines two mandatory subject confirmation methods:

- Holder-of-key
- Sender-vouches

The main XML elements of the SOAP message constructed according to [WSS Security] are depicted in Figure 2.

The SAML assertion is placed into <wsse:Security> header, which also contains the digital signature <ds:Signature>. The digital signature is used by the receiver of the SOAP message to verify that the sender of the message knows the key used for computing the signature over the digest of the SOAP body and for checking its integrity. The digest algorithm is SHA 1. The signature algorithm is RSA-SHA 1. The signature's value is provided in the <ds:SignatureValue> element of the <ds:Signature>.

Two subject confirmation methods define different ways for conveying information on the key to the receiver.



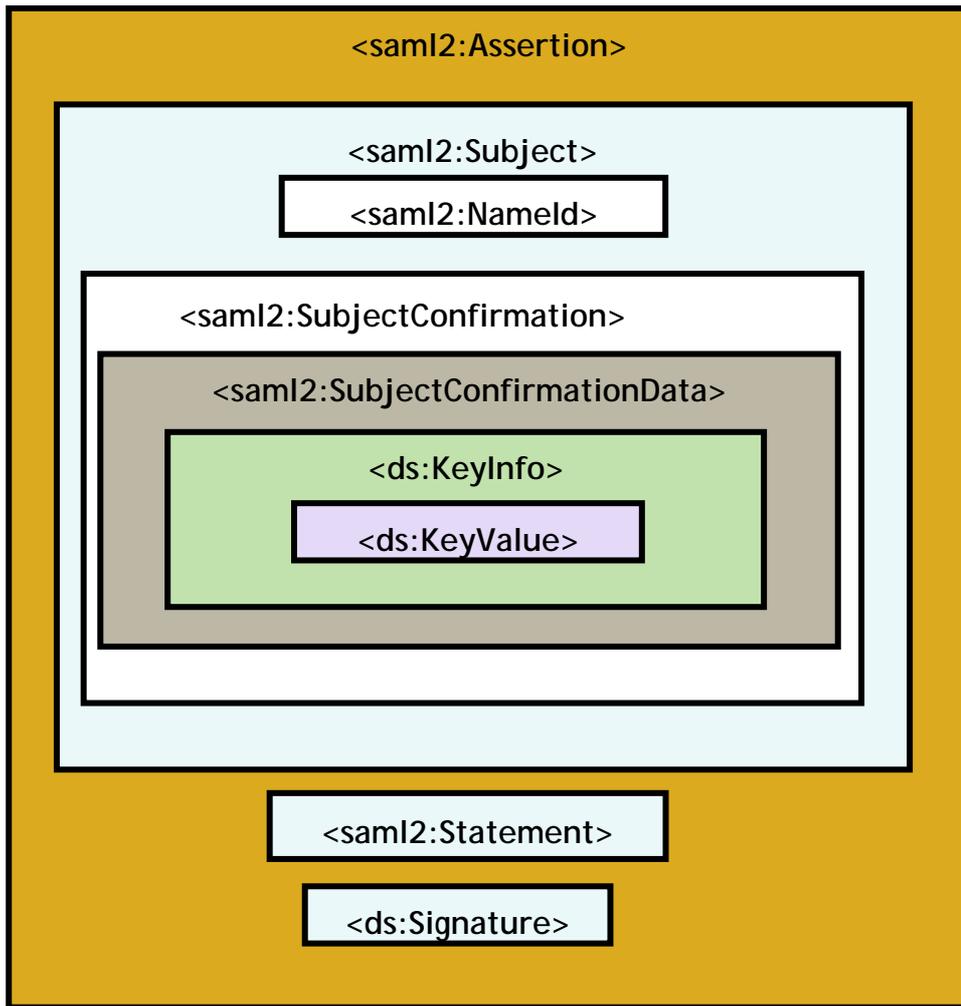
**Figure 2 - Structure of the SOAP message with SAML assertion**

The following clauses describe two subject confirmation methods.

#### **6.2.1.2.1 The holder-of-key subject confirmation method**

The Figure 3 depicts the structure of the SAML assertion used for holder-of-key subject confirmation method. The Method attribute of the element `<saml2:SubjectConfirmation>` indicates the method of the subject confirmation (holder-of-key).

The method specifies that the Sender (also known as Attesting Entity) must prove that it is entitled to make the Statements about the Subject by demonstrating knowledge of the key, which is identified in the `<ds:KeyValue>` element contained in the `<ds:KeyInfo>` element of the SAML assertion. The `<ds:KeyInfo>` element identifies a public or secret key that is used to confirm identity of the subject. The method further specifies that the sender may do it by signing a digest of the SOAP body with that key. The signature is contained in the element `<ds:Signature>` of the WS Security header as shown in Figure 2.



**Figure 3 - The structure of the SAML assertion used for the holder-of-key subject confirmation method**

The Receiver of the SOAP message obtains the key using information that is provided by the Attesting Entity (in `<ds:KeyInfo>`). The Receiver then computes digital signature of the SOAP body and checks whether it matches the signature provided by the Attesting Entity. If it is the case then the subject and statements of the SAML assertion may be attributed to the Attesting Entity and the content of the SOAP body whose integrity is protected by the key may be considered as provided by the Attesting Entity.

#### **6.2.1.2.2 The sender-vouches subject confirmation method**

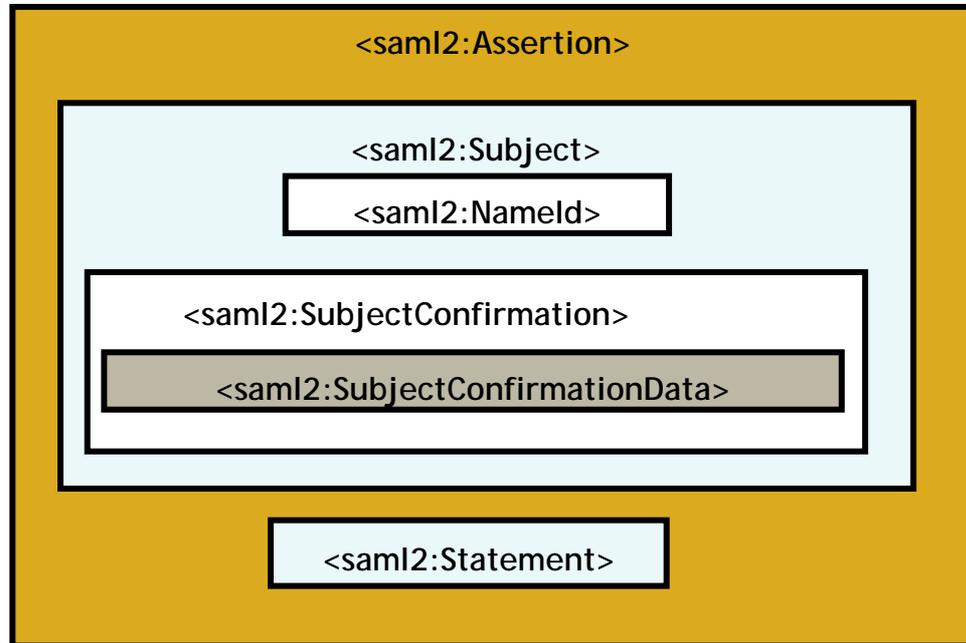
The Figure 4 depicts the structure of the SAML assertion used for sender-vouches subject confirmation method. The Method attribute of the element `<saml2:SubjectConfirmation>` indicates the method of the subject confirmation (sender-vouches).

The Attesting Entity is trusted by a Receiver to make SAML assertions regarding a subject under condition that value of the Method attribute of the `<SubjectConfirmation>` element indicates the sender-vouches method.

The Attesting Entity obtains one or more assertions or references to assertions from one or more authorities and includes them in a SOAP message. It then computes a signature of the digest of the SAML assertions and the body of the SOAP message. The signature is contained in the element

<ds:Signature> of the WS Security header (shown in Figure 2). The Attesting Entity optionally provides information to the Receiver on the key that was used to compute the signature. If there is no such information, the Receiver is expected to identify the key by other means.

The Receiver checks validity of the signature. If the signature is valid the Receiver establishes the fact that the statements have been made about the subject by the Attesting Entity.



**Figure 4 - The structure of the SAML assertion used for the sender-vouches subject confirmation method**

## 6.2.2 Certificate-based authentication

**Editor's note:** The above note is deleted and addressed by a proposed reference below.

The use of X.509 certificates for authentication is described in ITU-T Recommendation Y.2704, Security mechanisms and procedures for NGN.

## 6.2.3 Password-based authentication

**Editor's note:** The material of the contribution 1257 has not been included in (Y.2704). Consider removing this clause.

## 6.2.4 One-time Password

Depending on a required level of assurance OTP may be used. One method of implementing the OTP is described in [IETF RFC 2289].

## 6.2.5 Authentication and Key Agreement (AKA)

### 6.2.5.1 Overview of the AKA

The Universal Mobile Telecommunications System (UMTS) Authentication and Key Agreement (AKA) protocol supports mutual authentication of the Mobile Station (MS) and the network. The UMTS AKA is a challenge-response protocol, which uses a long-term key  $K$  shared between Universal Subscriber Identity Module (USIM) and Authentication Center (AuC), which could be a component of IdP. These entities reside on the Universal Integrated Circuit Card (UICC) of the MS and in the mobile station's home network respectively. The AKA protocol is specified in [ATIS33102].

The following acronyms are used in this clause:

AK	Anonymity Key
AKA	Authentication and Key Agreement
AMF	Authentication Management Field
AuC	Authentication Center
AUTN	Authentication token
AV	Authentication Vector
CK	Cipher Key
GPRS	General Packet Radio Services
HLR	Home Location Register
HSS	Home Subscriber Server
IK	Integrity Key
IMSI	International Mobile Subscriber Identity
ISIM	IP Multimedia Services Identity Module
MAC	Message Authentication Code
MS	Mobile Station
RAND	Random challenge
SQN	Sequence number
SGSN	Serving GPRS Support Node
UICC	Universal Integrated Circuit Card
UMTS	Universal Mobile Telecommunications System
USIM	Universal Subscriber Identity Module – an application on UICC
VLR	Visitor Location Register
XRES	Expected Response

The following entities are involved in the AKA authentication procedure:

- MS capable of running USIM application - a party to authentication procedure

- VLR/SGSN - another party to authentication procedure. It authenticates MS by comparing the response RES (received from the MS) and the expected response XRES (received from the AuC/HLR).
- AuC/HLR - an entity that shares a secret (K) with MS and provides VLR/SGSN with authentication vector (AV), which contains values used by VLR/SGSN for challenging the MS. The AV also provides the expected response (XRES), which is used for verifying the MS's response to the challenge.

The AKA authentication procedure begins with VLR/SGSN requesting the user identity. The USIM application responds with the user's IMSI – a unique identifier allocated to each mobile subscriber in an UMTS network. The details of the steps that follow are depicted by Figure 1 and described in the Appendix IV.

### 6.2.5.2 Generation of the authentication vector by AuC/HLR

The AuC/HLR starts generation of the authentication vector with generating a fresh sequence number SQN and an unpredictable challenge RAND, which are input parameters for calculation of the authentication vector AV. The SQN number is used for protection against a replay attack. To enable such protection the MS and AuC keep track of the SQN numbers that have been used. Additional information on SQN is provided in the clause *AKA Operation in USIM*.

An authentication vector AV is computed according to the following formulas:

- $AV = RAND || XRES || CK || IK || AUTN$
- $XRES = f_{2K}(RAND)$
- $MAC = f_{1K}(SQN || RAND || AMF)$
- $CK = f_{3K}(RAND)$
- $IK = f_{4K}(RAND)$
- $AK = f_{5K}(RAND)$
- $AUTN = SQN \oplus AK || AMF || MAC$ .

In these formulas  $f_{1K}()$ ,  $f_{2K}()$ ,  $f_{3K}()$ ,  $f_{4K}()$ , and  $f_{5K}()$  denote the functions, which generate the values with using the long-term secret key K. These functions are operator-specific. The symbols  $\oplus$  and  $||$  denote the exclusive OR (XOR) and concatenation operations respectively. The Authentication Management Field (AMF) is a 16-bit value, which is not a standardized value, although the specification suggests the examples of its use to:

- Indicate the authentication algorithm and key used to generate a particular authentication vector
- Change parameters for verification of freshness of the sequence number SQN (e.g., the acceptable range for the SQN)
- Restrict the lifetime of the cipher and integrity keys.

The Anonymity Key AK is used to conceal the SQN, which can be used to determine identity and location of the user.

### 6.2.5.3 AKA operation in USIM

After receiving the RAND and AUTN values from the VLR/SGSN, the USIM application computes  $AK = f5_K(RAND)$ , and then retrieves SQN by calculating  $SQN = (SQN \oplus AK) \oplus AK$ . The application proceeds with calculating the expected XMAC =  $f1_K(SQN || RAND || AMF)$  and comparing this value with the MAC received in the authentication token AUTN from the VLR/SGSN. If this verification fails, the USIM abandons the authentication procedure and sends an appropriate error message to the VLR/SGSN, which shall initiate Authentication Failure Report to the HLR.

If the MAC verification succeeds, the USIM proceeds with checking the freshness of the sequence number SQN. This checking procedure enables the USIM to detect the replayed messages. The procedure is based on comparison of the recently received sequence number with the sequence numbers that the USIM has previously received. The details of the verification of the sequence numbers' freshness by the USIM and their generation by the AuC are specified in [ATIS33102].

If the USIM determines that the SQN is not fresh, it sends the synchronization failure message to the VLR/SGSN and abandons the authentication procedure. The synchronization failure message contains information that enables the VLR/SGSN to initiate the re-synchronization procedure.

If the USIM determines that the SQN is fresh, it computes  $RES = f2_K(RAND)$  and includes it in the user authentication response. Finally, the USIM computes the confidentiality and integrity keys as follows:  $CK = f3_K(RAND)$  and  $IK = f4_K(RAND)$ .

### 6.2.5.4 Sizes of the AKA cryptographic parameters

The cryptographic parameters, which are used in AKA have the following sizes:

- AMF 16 bits
- AUTN 128 bits
- CK 128 bits
- IK 128 bits
- K 128 bits
- MAC 64 bits
- RAND 128 bits
- RES, (XRES) 32-128 bits (variable)
- SQN 48 bits

### 6.2.5.5 Universal Subscriber Identity Module (USIM)

The USIM is an application on the UICC. It implements AKA algorithms and stores, along with other data (e.g., telephone book), the parameters that are used by the AKA procedures. Particularly, it stores:

- International Mobile Subscriber Identity (IMSI)
- A long-term authentication key (K) - a secret shared with the AuC

- A sequence number (SQN)
- Temporary (session) cipher and integrity keys (i.e., CK and IK)
- The parameters for limiting the lifetime of CK and IK

#### 6.2.5.6 Use of the AKA in non-wireless environment

Although the AKA mechanism is typically used for authentication of the wireless devices that are equipped with the smart cards (e.g., UICC), there is nothing in the AKA specifications that would prevent the use of the mechanism for authentication of the fixed devices that are capable of running the USIM application.

**Editor's note:** The pros and cons for selecting AKA mechanism and guidelines for its use will be provided later.

#### 6.2.6 Integration of PKI-based authentication with IMS

IMS security is based on the AKA mechanism, which uses a shared secret and a challenge-response protocol for user-network authentication. But security of certain NGN services (e.g., IPTV) is based on PKI certificates. To ease blending of these NGN services and IMS services, it is desirable to integrate PKI-based authentication with the IMS and to do it in such a way that leverages the strength of IMS security.

Integration of the IMS with PKI-based authentication allows the user equipment and network to authenticate each other based on respective certificates and to agree on a set of cryptographic keys based on the same key generation algorithms as in AKA. To this end, the user equipment and network need to be provisioned with the respective private keys and certificates, and be able to perform the PKI operations.

With respect to agreement on the Ciphering Key (*CK*) and Integrity Key (*IK*) the described mechanism for integration specifies two options:

1. Establishing agreement on the *CK* and *IK* keys with the use of a shared secret between the End-User Function and S-5 - Service user profile functional entity (SUP-FE)
2. Establishing agreement on the *CK* and *IK* keys without the use of the shared secret

The generic call flow for the first option is depicted by Figure 5 and for the second option by Figure 6.

##### 6.2.6.1 Conventions

The following connections are used in this section:

“|” designates the string concatenation

*CK* designates Ciphering Key

*IK* designates Integrity Key

*K*(*·*) designates a symmetric key encryption

$N_{pr} [ ]$  designates encryption with the network private key  $N_{pr}$

$N_{pu} [ ]$  designates encryption with the network public key  $N_{pu}$  available from the network certificate

$U_{pr} [ ]$  designates encryption with the user private key  $U_{pr}$

### 6.2.6.2 Entities involved in authentication

- S-5 - Service user profile functional entity (SUP-FE)
- End-User Function. The entity is capable of running a SIP client
- S-n call session control functional entity (CSC-FE), where S-n stands for one of the following entities:
  - S-1 Serving call session control functional entity (S-CSC-FE)
  - S-2 Proxy call session control functional entity (P-CSC-FE)
  - S-3 Interrogating call session control functional entity (I-CSC-FE)

The S-n is used to denote one of these entities when there are no differences between them as far as the described authentication procedure is concerned.

### 6.2.6.3 Establishing agreement on the CK and IK keys with the use of a shared secret between the End-User Function and S-5 (option 1)

The call flow is depicted by Figure 5. The basic steps are as follows:

1. End-User Function sends SIP Register request with the user's *IMPU* and *IMPI* to the S-n
2. The S-1 requests a random challenge *RAND*, *CK*, and *IK* from the S-5. The values *RAND*, *CK*, and *IK* are specified in [ATIS33102]
3. The S-5 receives *RAND*, *CK*, and *IK* from the S-5 for the user
4. The S-n sends to the End-User Function the SIP Unauthorized message with a challenge *RAND* and its encrypted value  $N_{pr}[RAND]$

The End-User Function:

- Receives values *A*, which is supposedly equal to *RAND*, and *B*, which is supposedly equal to  $N_{pr}[RAND]$
  - Retrieves the network public key  $N_{pu}$
  - Decrypts *B* with  $N_{pu}$  and compares the result to *A*. If the values are equal, then the network is authenticated, if not – the authentication procedure is aborted
  - Generates *IK* and *CK* using the shared secret  $K_s$
  - Generates value  $U_{pr}[N_{pu}[K]/K(RAND)]$
5. End-User Function sends to the S-n SIP Register message with the *IMPU* and *IMPI* identifiers and the value  $U_{pr}[N_{pu}[K]/K(RAND)]$
  6. The S-1 sends to the S-5 data received in step 5 and requests verification and the user's record

The S-5 performs the following operations:

- Looks up the user certificate to obtain the user public key  $U_{pu}$
- Decrypts with  $U_{pu}$  the received value *C*, which is supposedly equal to  $U_{pr}[N_{pu}[K]/K(RAND)]$  to retrieve value *D/E*, where *D* is supposedly equal to  $N_{pu}[K]$  and *E* is supposedly equal to  $K(RAND)$
- Decrypts with the network private key  $N_{pr}$  value *D* to obtain *K'*

- Decrypts with  $K'$  value  $E$  to obtain  $RAND'$
  - Compares  $RAND'$  with  $RAND$ . If they match, the user has been authenticated
7. The S-5 communicates the authentication result and the user's record to the S-1
  8. The S-1 uses the record to check whether the authenticated user is authorized to register and receive the requested service. If that is the case, the S-n notifies the End-User Function that access is granted

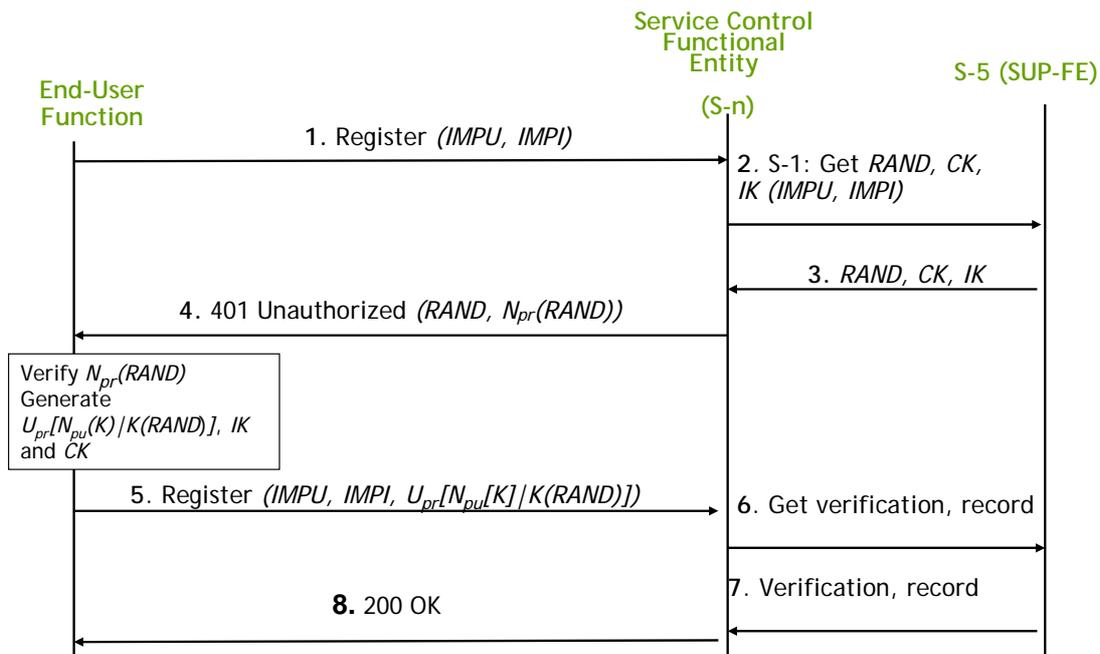


Figure 5 - Integration of the IMS authentication mechanism with PKI-based authentication (option 1)

#### 6.2.6.4 Establishing agreement on the CK and IK keys without the use of a shared secret between the End-User Function and S-5 (option 2)

The call flow is depicted by Figure 6. The basic steps are as follows:

1. End-User Function sends SIP Register request with the user's  $IMPU$  and  $IMPI$  to the S-n
2. The S-1 requests a random challenge  $RAND$  from the S-5. The value  $RAND$  is specified in [ATIS33102]
3. The S-1 receives  $RAND$  from the S-5 for the specified user
4. The S-n sends to the End-User Function the SIP Unauthorized message with a challenge  $RAND$  and its encrypted value  $N_{pr}[RAND]$

The End-User Function:

- Receives values  $A$ , which is supposedly equal to  $RAND$ , and  $B$ , which is supposedly equal to  $N_{pr}[RAND]$
- Retrieves the network public key  $N_{pu}$

- Decrypts  $B$  with  $N_{pu}$  and compares the result to  $A$ . If the values are equal, then the network is authenticated, if not – the authentication procedure is aborted
  - Generates  $IK$  and  $CK$  using the randomly-generated key  $K$
  - Generates value  $U_{pr}[N_{pu}[K]/K(RAND)]$
5. End-User Function sends to the S-n SIP Register message with the  $IMPU$  and  $IMPI$  identifiers and the value  $U_{pr}[N_{pu}[K]/K(RAND)]$
  6. The S-1 sends to the S-5 data received in step 5 and requests verification, the user's record, and the  $CK$  and  $IK$  keys

The S-5 performs the following operations:

- Looks up the user certificate to obtain the user public key  $U_{pu}$
  - Decrypts with  $U_{pu}$  the received value  $C$ , which is supposedly equal to  $U_{pr}[N_{pu}[K]/K(RAND)]$  to retrieve value  $D/E$ , where  $D$  is supposedly equal to  $N_{pu}[K]$  and  $E$  is supposedly equal to  $K(RAND)$
  - Decrypts with the network private key  $N_{pr}$  value  $D$  to obtain  $K'$
  - Decrypts with  $K'$  value  $E$  to obtain  $RAND'$
  - Compares  $RAND'$  with  $RAND$ . If they match, the user has been authenticated and  $K' = K$ . That is the End-User Function and S-5 now share key  $K$
  - Generates the  $CK$  and  $IK$  keys using the shared key  $K$ . For instance, the same functions for generating the  $CK$  and  $IK$  that are specified in [ATIS33102] can be employed with the use of  $K$  as an input parameter
7. The S-5 communicates the authentication result, the user's record, and the  $CK$  and  $IK$  keys to the S-1
  8. The S-1 uses the record to check whether the authenticated user is authorized to register and receive the requested service. If that is the case, the S-n notifies the End-User Function that access is granted

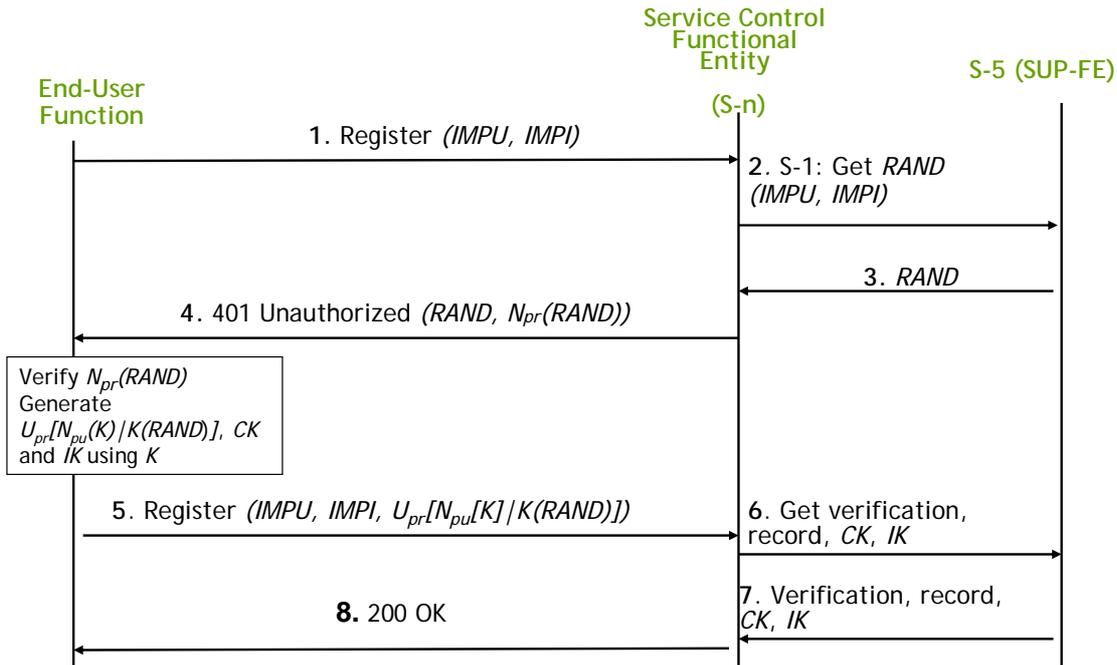


Figure 6 - Integration of the IMS authentication mechanism with PKI-based authentication (option 2)

### 6.2.6.5 Comparison of the option 1 and option 2

The described in options 1 and 2 mechanisms do not use a shared secret for authentication. While both options support integration of the PKI-based authentication and IMS, each has its advantages and disadvantages. Comparison between them is summarized in Table 1.

Table 1 – Comparison of the option 1 and option 2 for the key agreement between the End-User Function and S-5 on the CK and IK keys

	Option 1 (with pre-shared secret)	Option 2 (without pre-shared secret)
Advantages	Completely re-uses the AKA mechanism for establishing agreement on the CK and IK keys	Does not require provisioning of the shared secret between the End-User Function and S-5
Disadvantages	Requires provisioning of the shared secret between the End-User Function and S-5	Requires modifications to the applications running on the End-User Function (e.g., on a smart card) and S-5 for enabling agreement on the CK and IK

Option 1 should be selected to simplify key agreement on the CK and IK keys when the End-User Function and the S-5 share a secret. Option 2 should be the choice when the End-User Function and the S-5 do not have a shared secret.

The implementations of the proposed integration mechanism must support both options.

### Requirements on the S-5 functional entity

In addition to the capabilities specified in [ATIS33102], the S-5 must be capable of:

- Storing the users and network certificates
- Performing PKI-based decryption as described in step 6 (for both options)
- Running the Diameter protocol modified to carry information described in step 6 (for both options) and information needed for negotiation with the End-User Function on the PKI-based authentication
- Negotiating with the End-User Function an agreement on the PKI-based authentication method

#### 6.2.6.6 Requirements to the End-User Function

The End-User Function must be capable of:

- Securely storing the user's private key  $U_{pr}$
- Securely storing the shared secret  $K_s$  with the network (only for option 1)
- Storing a network X.509 certificate with the network's public key  $N_{pu}$
- Randomly generating one-time session key  $K$  and performing the symmetric key encryption with  $K$
- Generating the  $CK$  and  $IK$  keys with the use of the shared secret  $K_s$  as specified in [ATIS33102] (only for option 1)
- Generating the  $CK$  and  $IK$  keys as described in step 6 for option 2
- Performing PKI-based encryption and decryption described in Steps 4 and 5 for both options
- Running a SIP client with a modified SIP protocol enabling communication of information described in steps 4 and 5
- Negotiating with the S-2 an agreement on the use of the PKI-based authentication

#### 6.2.6.7 Requirements to the S-1

The additional requirements to the S-1 are as follows:

- It must be capable of constructing the SIP messages with information described in step 4 (for both options)
- It must be capable of retrieving from the SIP messages information described in step 5 and repackaging it into the Diameter messages as described in step 6 (for both options)
- It must be capable of performing the PKI-based encryption described in step 4 (for both options)
- It must be able to understand the notification from the S-5 on the use of the PKI-based authentication

### **6.2.6.8 Requirements on the SIP interfaces between the participating entities**

The End-User Function and the S-1 communicate via the S-2 and S-3 functional entities. The S-2 and S-3 entities are not essential to the described authentication and not shown in Figure 5 and Figure 6.

There are SIP interfaces between:

- End-User Function and S-2
- S-2 and S-3
- S-1 and S-3

These interfaces must be able to negotiate the use of the PKI-based authentication (including the specific option for key generation) and to carry information described in steps 4 and 5 (for both options).

### **6.2.6.9 Requirements on the Diameter interfaces between the participating entities**

There are Diameter interfaces between:

- S-1 and S-5
- S-3 and S-5

These interfaces must be able to negotiate the use of the PKI-based authentication (including the specific option for key generation) and to carry information described in step 6 (for both options)

**Editor's note:** The reference has been moved.

### **6.2.7 Integration of the PKI-based authentication and the SAML assertion mechanisms**

It is desirable that the application servers are able to provide their services to the clients that employ diverse authentication methods. The use of SAML allows achieving such an objective. Specifically, SAML allows having one entity (e.g., IdP) to perform authentication and another entity (e.g., application server) to use the authentication results. In such scenario IdP may implement the multiple authentication methods while the application servers rely on the IdP's SAML assertions. This scenario is beneficial to both the IdPs and the application providers. The benefits of the Application Service Providers (ASPs) are as follows:

- ASP may expand its customer base by providing services to the clients with various authentication methods
- ASP does not have to implement numerous authentication methods

IdP has the following benefits:

- It can attract more ASPs by offering its IdM services, particularly authentication
- IdP (especially when IdP is an NGN operator) can utilize its deployed authentication infrastructure

This clause specifies a mechanism for authenticating a client with the use of SAML assertions and PKI-based authentication. The mechanism, along with the one described in clause 6.2.6 *Integration*

of *PKI-based authentication with IMS*, allows the NGN operators to utilize their PKI-based infrastructure.

The mechanism is based on the SAML *HTTP redirect binding* specified in [X.1141].

### Entities involved in the authentication and the information flow

- End-User Function. This entity is capable of running a Web client and supporting PKI-based authentication [X.509].
- Application Server (AS) — an entity providing a Web service. It plays a role of a Relying Party. It acts as a SAML requestor as defined in [X.1141].
- A-2: Application gateway functional entity (APL-GW-FE), which is enabled to perform the PKI-based authentication and act as a SAML responder as defined in [X.1141].
- S-5 - Service user profile functional entity (SUP-FE)

The information flow of the authentication procedure is depicted by Figure 1 and described below.

### Conventions

The description uses the following conventions:

“|” designates the string concatenation

$K()$  designates a symmetric key encryption

$K_s$  designates a secret shared between A-2 and AS

$N_{pr} []$  designates encryption with the network private key  $N_{pr}$

$N_{pu} []$  designates encryption with the network public key  $N_{pu}$  available from the network certificate

$U_{pr} []$  designates encryption with the user private key  $U_{pr}$

$RAND$  designates randomly-generated challenge

### Mechanism's parameters

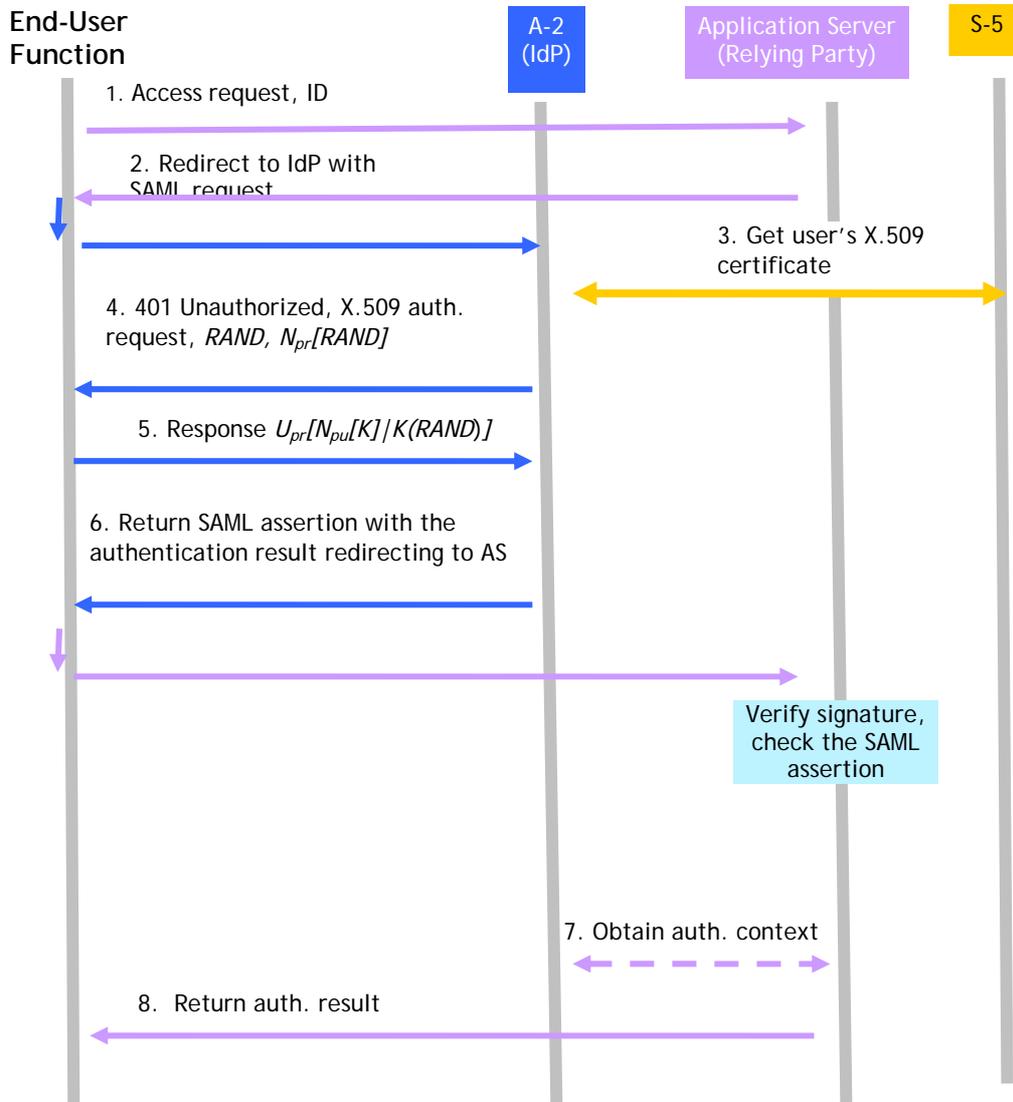
This clause specifies the mechanism-specific parameters. The list of the parameters is as follows:

`pki-auth-challenge` – parameter for transmitting the value of  $RAND$

`pki-auth-challenge-encrypted` – parameter for transmitting the value of  $N_{pr}[RAND]$

`pki-auth-user-signature` – parameter for transmitting the value of  $U_{pr}[N_{pu}[K]/K(RAND)]$

`pki-auth-keyinfo` – parameter for transmitting the value of  $K_s(K)$



### Figure 1 - The basic steps of data exchange for the PKI-based authentication with SAML-assertion

The mutual authentication of the End-User Function and A-2 is similar to the procedure used by the mechanism for integration of PKI-based authentication with IMS, which is described in clause 6.2.6.

The basic steps of the procedure that relies on the PKI-based authentication and SAML assertions are as follows:

9. A Web client of the End-User Function issues an HTTP request to the Application Server (AS). The request includes a user identifier and the URL of the A-2.
10. The Application server acting as a SAML requester responds to the HTTP request by sending a SAML request. The SAML request is encoded into the HTTP response's Location header with the HTTP status set to either 302 or 303. The agent of the End-User Function delivers the SAML request by issuing HTTP GET request to A-2, which acts as a SAML responder. This HTTP redirection procedure, known as *HTTP redirect binding*, is specified in [X.1141]. To ensure authentication and integrity of the URL-encoded message it should be signed as specified in clause 10.2.4.5.2 *Security Considerations* of [X.1141]. The shared secret  $K_s$  must be used for signing.
11. After signature validation, A-2 obtains from the S-5 the End-User's certificate and checks whether it is valid. The certificate contains the End-User Function's public key.
12. A-2 responds to the End-User Function with an HTTP response message indicating that authentication with the use of X.509 certificate is required. This is accomplished by setting the value of the response header WWW-Authenticate [RFC 2616] to "pki-auth". The body of the message includes the pki-auth-challenge and pki-auth-challenge-encrypted parameters that carry the values of the randomly-generated challenge  $RAND$  and its encryption  $N_{pr}[RAND]$  respectively. The header Content-Type must be set to application/x-www-form-urlencoded.
13. The End-User Function
  - Retrieves values  $A$ , which is supposedly equal to  $RAND$ , and  $B$ , which is supposedly equal to  $N_{pr}[RAND]$
  - Retrieves the network public key  $N_{pu}$
  - Decrypts  $B$  with  $N_{pu}$  and compares the result to  $A$ . If the values are equal, then the network is authenticated, if not – the authentication procedure is halted
  - Generates a secret key  $K$
  - Generates value  $U_{pr}[N_{pu}[K]/K(RAND)]$ , sets the parameter pki-auth-user-signature to that value, and sends it in the body of an HTTP POST message to A-2. The header Content-Type of the message must be set to the value application/x-www-form-urlencoded

After this step the A-2 checks whether the response is valid. To that end A-2 performs the following operations:

- Looks up the user certificate to obtain the user public key  $U_{pu}$
- Decrypts with  $U_{pu}$  the received value  $C$ , which is supposedly equal to  $U_{pr}[N_{pu}[K]/K(RAND)]$  to retrieve value  $D/E$ , where  $D$  is supposedly equal to  $N_{pu}[K]$  and  $E$  is supposedly equal to  $K(RAND)$

- Decrypts with the network private key  $N_{pr}$  value  $D$  to obtain  $K'$
- Decrypts with  $K'$  value  $E$  to obtain  $RAND'$
- Compares  $RAND'$  with  $RAND$ . If they match, the user has been authenticated and  $K' = K$ . That is the End-User Function and A-2 now share key  $K$

14. If all the above steps were successful, the A-2 performs the following operations

- Generates a SAML assertion setting the attribute Method of the element `<SubjectConfirmation>` to the value `sender-vouches`.
- Computes value  $K_s(K)$ .
- Includes the assertion in a SAML response. It then delivers the SAML response and the computed value  $K_s(K)$  over HTTP in the same manner as described for the SAML request in step 2 (i.e., as part of a query string). The value  $K_s(K)$  is carried by the parameter `pkinfo-auth-keyinfo`
- To ensure origin authentication and integrity of the URL-encoded message, A-2 signs it as specified in clause 10.2.4.5.2 *Security Considerations* of [X.1141]. The shared secret  $K_s$  should be used for signing.

After validating the signed URL, AS is assured that the SAML assertion is made by A-2. The AS checks the assertion itself (e.g., to ensure that *conditions* are met). After that, the AS retrieves the value  $K_s(K)$  and decrypts it using shared  $K_s$  to obtain  $K$ . At this point AS has authenticated the End-User Function and both entities share the key  $K$ , which can be used for securing communications between them.

15. The AS, if it is required by policy for making an authorization decision, obtains information about the authentication context. In that case A-2 responds with information specified by the *Public key – X.509* authentication context class [X.1141].
16. AS sends to the End-User Function the result of the authorization decision.

### **Additional requirements for the entities participating in the authentication**

In order to support the described mechanism, the participating entities must meet the following requirements:

- Requirements for the End-User Function  
The End-User Function must be capable of:
  - Running HTTP client
  - Securely storing its private key  $U_{pr}$  (e.g., on a smart card)
  - Obtaining the network public key  $N_{pu}$
  - Performing encryption and decryption
  - Generating a key  $K$
- Requirements for the Application Server (AS)
  - AS must support SAML [X.1141]
  - AS must have a shared secret ( $K_s$ ) with A-2
- Requirements for the A-2 Functional Entity

The A-2 functional entity must be able to:

- Support HTTP protocol
- Securely store its private key  $N_{pr}$
- Obtain the user public key  $U_{pu}$
- Perform encryption and decryption
- Generate a random challenge  $RAND$
- Support SAML [X.1141]
- Have a shared secret ( $K_s$ ) with AS
- Requirements for the S-5 Functional Entity

The S-5 Functional Entity must securely store the users' X.509 certificates

### **Additional requirements for the interfaces between the participating entities**

The requirements for the interfaces are as follows:

- The interface between the End-User Function and the Application Server must support HTTP protocol [RFC 2616]
- The interfaces between the End-User Function and A-2 Functional Entities must support HTTP protocol [RFC 2616]
- The interface between A-2 and Application Server must support SAML [X.1141]
- The interface between the A-2 and S-5 Functional Entities must support a query-response mechanism that allows A-2 to obtain the users' X.509 certificates from S-5

### **6.2.8 Integration of *OpenID*-based authentication with IMS**

The mechanism, which allows integration of the IMS- and *OpenID*-based authentication combines capabilities of the *network-centric* IdM with those of the *user-centric*. The mechanism:

- Enables the network operators to provide identity services to the users accessing the Web applications
- Provides users with an SSO across the IMS and web services with an existing ISIM application
- Allows users to control their public identifiers on the Web as specified in [OpenID]
- Improves user security by engaging a user-trusted network operator in the access control to the Web applications

#### **6.2.8.1 Entities involved in the authentication and the information flow**

- End-User Function. This entity is capable of running a Web client and communicating with the ISIM application
- Application server — an entity providing a Web service. It plays a role of a Relying Party

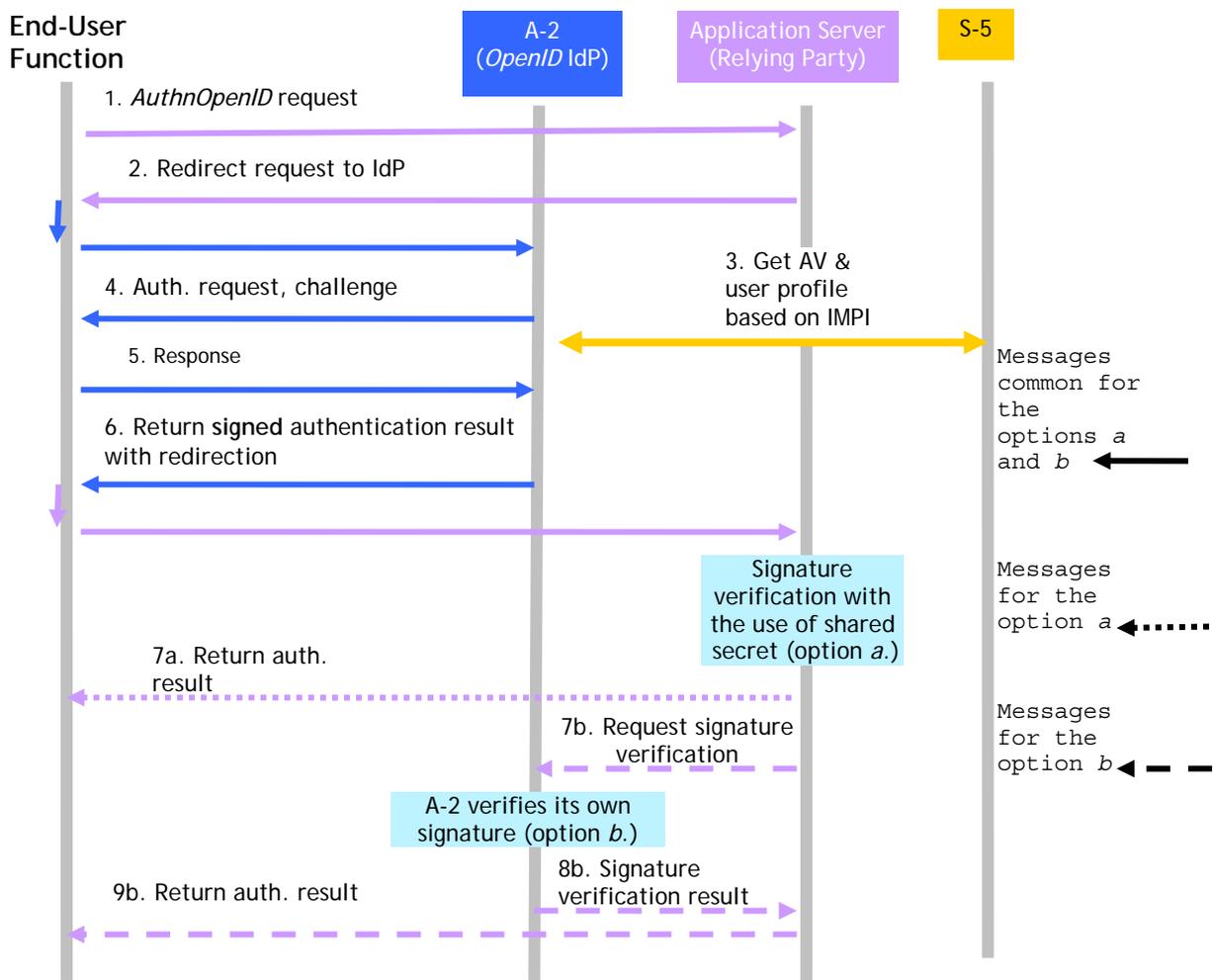
- A-2: Application gateway functional entity (APL-GW-FE), which is enabled to serve as an *OpenID* [OpenID] identity provider. (The A-2 optionally shares a short-term secret with the Application server as specified in [OpenID])
- S-5 - Service user profile functional entity (SUP-FE)

The information flow of the authentication procedure is depicted by Figure 7. The procedure of establishing the short-term signing key between the Application server and A-2 is not shown. The figure shows the basic steps of the procedure for two *OpenID* options:

- The A-2 and the Application server share a secret
- The A-2 and the Application server do not share a secret

The common steps for both options are 1 through 6. The step 7a is for the option *a* only.

The steps 7b, 8b, and 9b are for the option *b* only.



**Figure 7 - Integration of the IMS authentication mechanism with *OpenID*-based authentication**

The basic steps are as follows:

1. A Web client of the End-User Function issues an authentication request *AuthnOpenID* to the Application server. The request includes an *OpenID* identifier.
2. The Application server, using the presented *OpenID* identifier, discovers the URL of the A-2, which serves as an *OpenID* identity provider, and redirects the user authentication request to that URL.

After this step the A-2 correlates the user identifier with the IMS private and public identifiers.

3. A-2 obtains from the S-5 the AKA authentication vector AV (described in clause 6.2.5.2) and the user profile based on the IMPI.
4. A-2 sends to the End-User Function authentication request using the HTTP Digest AKA method [RFC 4169] or [RFC 3310]. The request includes a challenge and a quantity that enables the End-User Function to authenticate the network.

After this step the End-User Function authenticates the network as specified in [RFC 4169] or [RFC 3310].

5. End-User Function sends to the A-2 response to the challenge as specified in [RFC 4169] or [RFC 3310].

After this step the A-2 authenticates the End-User Function as specified in [RFC 4169] or [RFC 3310].

6. The A-2 sends to the End-User Function a signed message asserting that the claimed *OpenID* identifier belongs to the user. The message is signed with the use of a secret shared with the Application server for the option a. For the option b the message is signed with the A-2 secret key. The message includes a request to redirect the Web client of the End-User Function to the Application server. The details of the signing and redirection procedures are described in [OpenID].

#### **Steps that are specific to option a:**

- 7a. After verifying the signature of the response received in step 6, the Application server notifies the End-User Function of the authentication result. The Application server uses the secret shared with the A-2 for such verification.

If there is a failure in one of the following steps: 1 through 6, or 7a – the authentication procedure stops.

#### **Steps that are specific to option b:**

- 7b. The Application server sends a copy of the message received in step 6 to the A-2 with a request to verify the signature.
- 8b. After verifying its own signature, the A-2 reports the verification result to the application server.
- 9b. The Application server reports the authentication result to the End-User Function.

If there is a failure in one of the following steps: 1 through 6, 7b, 8b, or 9b – the authentication procedure stops.

#### **6.2.8.2 Additional requirements for the entities participating in the authentication**

In order to support the described mechanism, the participating entities must meet the following requirements:

- Requirements for the End-User Function

The End-User Function must be capable of:

- Authenticating with the use of the HTTP Digest AKA method
- Communicating with ISIM application

- Requirements for the Application server

The Application server must be able to support *OpenID* specification version 2.0 [OpenID]

- Requirements for the A-2 Functional Entity

The A-2 functional entity must be able to:

- Perform HTTP Digest AKA authentication
- Correlate the user *OpenID* identifier with her or his IMS public and private identities
- Serve as an *OpenID* identity provider

- Requirements for the S-5 Functional Entity

There are no other requirements to the S-5 Functional Entity than those specified in [FRA]

### 6.2.8.3 Additional requirements for the interfaces between the participating entities

The requirements for the interfaces are as follows:

- The interface between the End-User Function and the Application server must support the *OpenID* authentication as specified in specification version 2.0 [OpenID]
- The interfaces between the End-User Function and A-2 Functional Entities must support the HTTP Digest AKA protocol [RFC 4169] or [RFC 3310]
- The interface between the A-2 and S-5 Functional Entities does not have any mechanism-specific requirements

**Editor's note:** References for 6.2.7; should be moved to the References section

[ATIS33102] ATIS.3GPP.33.102V710-2007, *Security Architecture*

[OpenID] *OpenID Authentication 2.0* <[http://openid.net/specs/openid-authentication-2\\_0.html](http://openid.net/specs/openid-authentication-2_0.html)>

[RFC 3310] IETF RFC 3310 (2002), *Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA)* <<http://www.rfc-editor.org/rfc/rfc3310.txt>>

[RFC 4169] IETF RFC 4169 (2005), *Hypertext Transfer Protocol (HTTP) Digest Authentication Using Authentication and Key Agreement (AKA) Version-2* <<http://www.ietf.org/rfc/rfc4169.txt?number=4169>>.

**Editor's note:**

Contributions are invited to address the following observations:

- It seems that the A-2 acts as a sort of GBA credential server (BSF) and the interface of A-2 with the end-user cannot be identical to Ub because OpenID requires the first message sent by the user to A-2 to be just the message redirected from the application server.
- The second and third messages from GBA Ub interface between terminal and BSF, could be identical to message 4 and 5 from the document. The interface between A-2 and the HSS could be a simple version of

Zh (interface between BSF and HSS). The Zn interface between application server and BSF is replaced with an interface A-2 - app server from OpenID, and there is no distribution of a shared key as in GBA.

- Additionally, there should be a security requirement on the interface between the application server and the user (A-2), else the user could be impersonated, if someone obtains the signed authentication result in step 6. A reference here to OpenID or directly to TLS could do the trick.
- Also it seems to be assumed in this system that the application server is trustworthy and would not impersonate the user with another application server. Here is a difference to GBA that does not need to make this assumption, due to the usage application specific key.

### 6.2.9 GBA

The Generic Bootstrapping Architecture (GBA) specifies a framework for bootstrapping authentication and establishing key agreement leveraging the 3GPP Authentication and Key Agreement (AKA) mechanism. The GBA facilitates authentication of the End-Users to Network Application Function (NAF) and can be used in NGN Identity Management for enabling:

- Authentication and key agreement
- Privacy protection
- Single Sign On

The GBA is an authentication system that includes three parties:

- An end-user who is trying to obtain network services using User Equipment (UE)
- Application server (called Network Application Function or NAF)
- A trusted entity (called Bootstrapping Server Function or BSF), which is involved in authentication and key exchange between two other entities.

The GBA enables authentication of the End-User, who is using UE, to an application server (NAF) without revealing the End-User's long-term credentials and secrets to the NAF by using a trusted entity BSF.

The basics of the GBA authentication process are illustrated by the reference model and described below. The following acronyms are used:

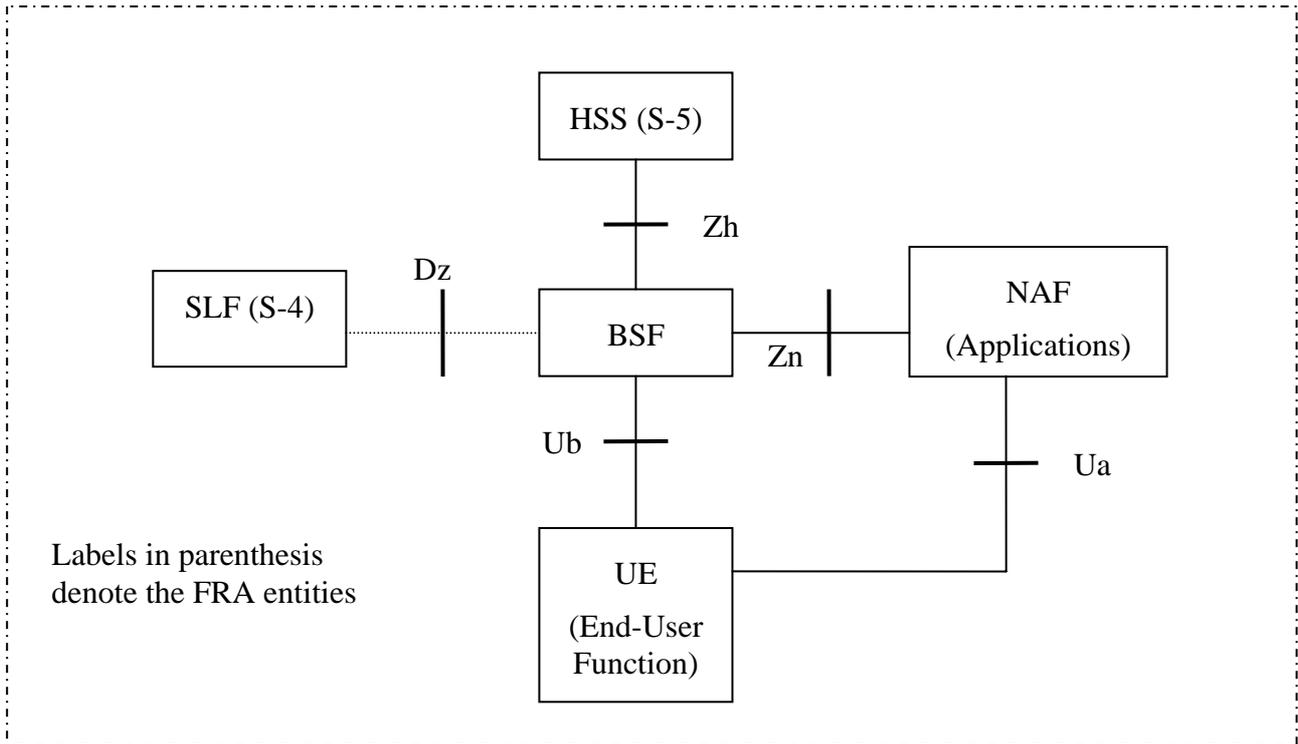
BSF    Bootstrapping Server Function

HSS    Home Subscriber System

NAF    Network Application Function

SLF    Subscriber Locator Function

UE     User Equipment



**Figure 8 – Simple network model for bootstrapping**

(Contributor's note: Figure 5 is reproduced from the standard ETSI TS 133 220)

These are the basic steps of the GBA procedure:

1. NAF requests authentication and negotiates the use of GBA over Ua reference point.
2. The BSF client that runs on the UE initiates bootstrapping procedure over the reference point Ub. The BSF fetches authentication information and the GBA user security settings from the HSS over Zh. The UE and the BSF mutually authenticate using http Digest AKA. The procedure results in the UE receiving bootstrapping transaction identifier (B-TID) from the BSF and establishing a shared key (Ks) between the UE and the BSF.
3. UE derives Ks\_NAF from Ks and sends B-TID (along with the application-specific data) to the NAF.
4. The NAF sends B-TID to the BSF over Zn reference point.
5. The BSF based on B-TID determines the Ks that should be used, derives Ks\_NAF from it and sends Ks\_NAF to the NAF.
6. Finally, UE and NAF can authenticate each other using the shared key Ks\_NAF. The exact authentication procedure depends on the protocol between the UE and NAF. For instance, GBA specifies that HTTP-based applications can use either HTTP Digest authentication (RFC 2617) or TLS pre-shared key ciphersuites (RFC 4279).

Note: The BSF queries the SLF over the Dz reference point to obtain the name of the HSS containing the subscriber-specific data. The SLF is not needed when the BSF is configured to use a pre-defined HSS.

Mapping of the GBA entities to the NGN entities specified in Y.2012, *Functional requirements and architecture of the NGN of Release 1*.

- NAF - corresponds to *Applications* entity of the Y.2012 Figure 3: NGN generalized functional architecture.
- BSF **Editor's note**: Presently, Functional Architecture, which is defined in Y.2012, does not specify an entity that corresponds to BSF. According to the experts of Q.5/13, additional studies are needed to introduce such an entity into the architecture. The experts of Q.5/13 have asked for contributions, which would justify the need of GBA for NGN.

**Editor's note**: Addition of FE with BSF capabilities was discussed by Q.5/13 at its May 2009 meeting. It was decided that there are no sufficient reasons for introducing such an entity. We should consider whether to keep a section on GBA in this document.

- HSS corresponds to S-5 Service User Profile FE
- SLF corresponds to S-4 Subscription Locator FE
- UE corresponds to the End-User Function

**Editor's note**: The editor proposes to remove this clause on GBA because there is no functional entity in FRA with the capabilities of BSF, and no sufficient reasons for introducing such an entity has been found.

### 6.3 Correlation and Binding

This section would recommend mechanisms to correlate and bind identity information (e.g., binding user and device identities)

### 6.4 Discovery

**Editor's note**: This section should recommend protocol and mechanisms to discovery identity information. This include identity information with a NGN provider network (e.g., information in location server, presence servers and HSS). It should also recommend protocols and mechanisms to discover identity information in a federated environment.

#### 6.4.1 Intra-network Discovery

**Editor's note**: This section would recommend protocol and mechanisms to discover identity information within NGN provider network (e.g., information in location server, presence servers and HSS).

#### 6.4.2 Inter-network Discovery

**Editor's note**: This section would recommend protocol and mechanisms to discover identity information across different NGN providers.

### 6.5 Policy Enforcement

**Editor's note**: This section would recommend procedures regarding the enforcement of applicable policies.

## 6.6 IdM Communications and Information Exchange

**Editor's note:** This section would recommend protocols and mechanisms to communicate and exchange identity information

### 6.6.1 External Interfaces

**Editor's note:** This section would recommend mechanisms and protocols to use across external interfaces (e.g., UNI, ANI/SNI and NNI) to communicate and exchange identity information.

- GBA-based mechanisms
- SAML-based mechanisms
- Other?

### 6.6.2 Internal Interfaces

**Editor's note:** This section would recommend mechanisms and protocols to use on internal interfaces to communicate and exchange identity information.

## 6.6.3 Security of IdM Communications and Exchange

**Editor's note:** This section recommends mechanisms to provide integrity and confidentiality protection of IdM communications

### 6.6.3.1 SAML 2.0 (ITU-T Recommendation X.1141 [4])

For both integrity and confidentiality protection, SAML 2.0 recommends the use of a secure channel or secure network protocol such as TLS or IPsec to be configured to protect the packets transmitted via the network connection.

For message level integrity protection in addition to the secured communication channel, XML Signature can be used. The section "8.4 SAML and XML signature syntax and processing" of ITU-T Recommendation X.1141 [4] is required to be followed when XML signature is used.

For message level confidentiality protection in addition to the secured communication channel, XML Encryption can be used. The section "8.4 SAML and XML signature syntax and processing" of ITU-T Recommendation X.1141 [4] is required to be followed when XML Encryption is used.

### 6.6.3.2 Identity Web Services Framework (known as ID-WSF)

The ID-WSF can be used for identity-based world wide web services. In order to use ID-WSF, its communications and its messages between the sender and recipient are expected to have their integrity and confidentiality protected. Like SAML 2.0, it recommends the use of a secure channel or secure network protocol such as TLS or IPsec to be configured to protect the packets transmitted via the network connection.

#### (1) Transport Layer Channel Protection

In case of using SSL or TLS as secure network protocol for ID-WSF, it is required to use either SSL 3.0, TLS 1.0 or higher. An entity that terminates an SSL (3.0) or TLS (1.0) connection is required to

offer or accept suitable cipher suites during the handshake. Recommended TLS 1.0 cipher suites (or their SSL 3.0 equivalent) are as follows, although they are not exhaustive.

- TLS\_RSA\_WITH\_RC4\_128\_SHA
- TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA
- TLS\_DHE\_DSS\_WITH\_3DES\_EDE\_CBC\_SHA
- TLS\_RSA\_WITH\_AES\_CBC\_SHA
- TLS\_DHE\_DSS\_WITH\_AES\_CBC\_SHA

For signing and verification of protocol messages, communicating entities is recommended to use certificates and private keys that are distinct from the certificates and private keys applied for SSL or TLS channel protection.

Other security protocols such as IPsec or Kerberos may be used as long as they implement equivalent security measures.

### (2) Message Confidentiality Protection

In the presence of intermediaries, communicating entities are required to ensure that sensitive information is not disclosed to unauthorized entities. In this case, these entities are required to use the confidentiality mechanisms specified in Web Services Security (WSS) SOAP Message Security by OASIS [6], to encrypt the SOAP envelope <S:Body> Content.

### (3) Message Integrity Rules

Message Integrity Rules in this section only applies if Web Services Security (WSS) SOAP Message Security by OASIS [6] is used for a ID-WSF protocol message bound to SOAP according to the Liberty SOAP Binding Version 2.0 [7].

In this case the sender is required to create a single <ds:Signature> contained in the <wsse:Security> header and this signature is required to reference all of the message components required to be signed.

In particular, this signature is required to reference the SOAP Body element (the element itself), the security token associated with the signature, and all headers in the message that have been defined in the Liberty SOAP Binding Version 2.0 [7], including both required and optional header blocks.

An example security token is a <saml2:Assertion> element conveyed in the <wsse:Security> header. The wsu:Timestamp header in the wsse:Security header block, the wsa:MessageID, wsa:RelatesTo, sb:Framework, sb:Sender and sb:InvocationIdentity header blocks are examples of header elements that would be referenced in a signature.

Note that care is required to be taken when constructing elements contained in Reference Parameters in Endpoint References, as these will be promoted to SOAP header blocks. Effort is recommended to be taken to avoid conflicting or duplicate id attributes, for example by using techniques to generate ids where it is highly likely that they are unique.

If the message is signed the sender is required to include the resultant XML signature in a <ds:Signature> element as a child of the <wsse:Security> header.

The <ds:Signature> element is required to refer to the subject confirmation key with a <ds:KeyInfo> element. The <ds:KeyInfo> element is required to include a <wsse:SecurityTokenReference> element so that the subject confirmation key can be located within the <wsse:Security> header. The inclusion of the reference is recommended to adhere to the guidance specified in section 3.4.2 of Web Services Security: SAML Token Profile 1.1. by OASIS [8].

i) Sender Processing Rules

- The construction and decoration of the <wsse:Security> header element is required to adhere to the rules specified in the Web Services Security: SAML Token Profile 1.1. by OASIS [8].
- The <wsse:Security> header element is required to have a mustUnderstand attribute with logical value true.
- The sender is required to place the message authentication security token as a direct child of the <wsse:Security> element.
- The sender is required to follow the Message Integrity rules outlined for senders and recipients when message authentication mechanisms are used.

The following considerations do not apply to Bearer tokens:

- For deployment settings which require independent message authentication, the obligation is required to be accomplished by signing the message body and portions of the header and placing the <ds:Signature> as a direct child of the <wsse:Security> header.
- For deployment settings which do not require independent message authentication then the subject confirmation obligation may be accomplished by correlating the certificate and key used to affect peer entity authentication with the certificate and key described by the message authentication token. To accommodate this, the assertion issuing authority is required to construct the assertion such that the confirmation key can be unambiguously verified to be the same certificate and key used in establishing peer entity authentication. This is necessary to mitigate the threat of a certificate substitution attack. It is recommended that the certificate or certificate chain be bound to the subject confirmation key.

ii) Recipient Processing Rules

- The recipient is required to locate the <wsse:Security> element for which it is the target. This MUST adhere to the rules specified in Web Services Security (WSS) SOAP Message Security by OASIS [6] and the applicable WSS token profiles (e.g., Web Services Security: SAML Token Profile 1.1. by OASIS [8] for SAML tokens).
- The <wsse:Security> header element is required to have a mustUnderstand attribute with logical value true and the recipient must be able to process this header block according to Web Services Security (WSS) SOAP Message Security by OASIS [6] and the appropriate WSS token profiles (e.g., Web Services Security: SAML Token Profile 1.1. by OASIS [8] for SAML tokens).
- The recipient is required to locate the security token and the recipient is required to determine that it trusts the authority which issued the token.
- The recipient is required to validate the issuer's signature over the token. This validation is required to conform to the core validation rules described in XML Signature Syntax and Processing (Second Edition) by World Wide Web Consortium (W3C) [9]. The recipient is recommended to validate the trust semantics of the signing key, as appropriate to the risk of incorrect authentication.
- If the message has been signed then the recipient is required to locate the <ds:Signature> element carried inside the <wsse:Security> header.
- Unless the security mechanism is peerSAMLV2 the recipient is required to resolve the contents of the <ds:KeyInfo> element carried within the <ds:Signature> and use the key it

describes for validating the signed elements. When the security mechanism is peerSAMLV2, the key is the client key used in SSL/TLS client authentication.

- The recipient is required to follow the Message Integrity rules outlined for senders and recipients when message authentication mechanisms are used.

#### (4) Processing messages with WSS X.509 token

The semantics and processing rules for mechanisms with MESSAGE having the value of X509 are described in this section. An example can be found at Appendix V.

These URIs support unilateral (sender) message authentication and are of the form:

- *urn:liberty:security:2003-08:PEER:X509* where PEER may vary depending on the peer authentication mechanism deployed (e.g., may be null, TLS etc).

The WSS X509 message authentication mechanism uses the Web Services Security X.509 Certificate Token Profile [2] as the means by which the message sender authenticates to the recipient. These message authentication mechanisms are unilateral. That is, only the sender of the message is authenticated. It is not in the scope of this recommendation to suggest when response messages should be authenticated but it is worth noting that this mechanism could be relied upon to authenticate the response message as well. It is recommended to recognize, however, that independent authentication of response messages does not provide the same message stream protection semantics as a mutual peer entity authentication mechanism would offer.

For deployment settings that require message authentication independent of peer entity authentication, then the sending peer is required to perform message authentication by demonstrating proof of possession of the key associated with the X.509 token. This key is required to be recognized by the recipient as belonging to the sending peer.

When the sender wields the subject confirmation key to sign elements of the message the signature ensures the authenticity and integrity of the elements covered by the signature. However, this alone does not mitigate the threat of replay, insertion and certain classes of message modification attacks. To secure the message from such threats, one of the mechanisms which support peer entity authentication can be used or the underlying SOAP binding request processing model is required to address these threats.

##### i) Sender Processing Rules

The rules in this section are in addition to the generic message authentication processing rules specified in this document.

The sender is required to demonstrate possession of the private key associated with the signature generated in conjunction with the WSS X509 token profile.

For deployment settings which REQUIRE independent message authentication, the obligation is required to be accomplished by signing portions of the message as appropriate and recording information in the <wsse:Security> header (as outlined in [4]).

For deployment settings which DO NOT REQUIRE independent message authentication then the sender is required to accomplish this obligation by decorating the security header with a <ds:KeyInfo> element bearing the certificate.

This is required to be unambiguously verified to be the same certificate and key used in establishing peer entity authentication. This is necessary to mitigate the threat of a certificate

substitution attack. Also note that this optimization only applies to *ClientTLS:X509* mechanisms.

ii) Recipient Processing Rules

If the validation policy regards peer entity authentication sufficient for purposes of authentication then the recipient is required to establish the correspondence of the certificate and key used to establish peer authentication with the corresponding key information conveyed in the message. This allows the message recipient to determine that the message sender intended a particular transport authenticated identity to be used. Information relating the SSL/TLS key to the message MAY be conveyed in the message using an OASIS SOAP Message Security X.509 security token.

**Editor's note:** these references should go to the References section. All of them are normative for implementation of the described in this clause mechanisms. The implementation is optional.

[2] Web Services Security X.509 Certificate Token Profile 1.1, OASIS

[3] Liberty ID-WSF Security Mechanisms Core, Liberty Alliance Project

[4] Web Services Security (WSS) SOAP Message Security, OASIS

**Editor's note:** the above references are for (4) *Processing messages with WSS X.509 token*

[4] ITU-T Recommendation X.1141

[5] Liberty ID-WSF Security Mechanisms Core

[6] Web Services Security (WSS) SOAP Message Security, OASIS

[7] Liberty SOAP Binding Version 2.0, Liberty Alliance Project

[8] Web Services Security (WSS) SAML Token Profile 1.1. OASIS

[9] XML Signature Syntax and Processing (Second Edition) by World Wide Web Consortium (W3C)

## 6.7 User and Subscriber Control

This section would recommend mechanisms and procedures regarding user control of their private information (e.g., permission for dissemination of personal information, and delegation)

## 6.8 Protection of Personally Identifiable Information (PII)

This section would recommend mechanisms and procedures related to protection of PII.

## 6.9 Federated Identity Functions

This section will recommend, mostly through reference solution for federated identity functions

## 6.9.1 Bridging and Interworking

**Editor's note:** This section will recommend mechanisms and procedures to allow bridging and interworking between different federations and IdM solutions. For example IdM solutions using different schemas and representations for identity information.

## 6.9.2 Discovery of IdPs in Federated Environment

**Editor's note:** This clause will describe the Push and pull mechanisms for sharing authentication information among IdPs

## 6.10 Identity Information Access Control

### 6.10.1 SAML-based mechanism for attribute sharing

The attribute sharing can be done with the use of the SAML assertions containing the attribute statements. The mechanism described in section 6.2.1 can be used for distributing of the SAML tokens.

### 6.10.2 Pseudonym management

The IDP can generate the pseudonym mapping to identity when receiving pseudonym generation request.

The IDP can query the relationship between pseudonym and identity to get the identity based on the pseudo identifier when receiving identity querying request.

The requesting entity accesses the service using pseudonym and information which is signed by private secret key, which is related to the pseudonym identifier and indicates the requesting entity's identity.

[**Editor's note:** Shin will provide relevant references to SAML; **Editors note:** Shin's References: Security Assertion Markup Language (SAML) V2.0 Technical Overview Committee Draft 02 25 March 2008

<<<http://www.oasis-open.org/committees/download.php/27819/sstc-saml-tech-overview-2.0-cd-02.pdf>>> **Editor's note:** Need to find out whether a draft can be referenced.

and ITU-T Recommendation X.1141 Security Assertion Markup Language (SAML)]

- Static pseudonyms
- Dynamic pseudonyms

### 6.10.3 Integrity of the SAML assertions

- Subject confirmation methods of the SAML tokens

## 6.11 Single Sign-on

Single sign-on (SSO) is a network capability that enables a user to log in once and obtain access to the multiple resources of a network without being repeatedly requested to provide her or his authentication credentials. This capability significantly improves user experience by enabling a user

to receive various services without having to maintain multiple authentication credentials (e.g., username/password pairs). In the environment where the users have to maintain multiple user names and passwords, they tend to create the simple passwords, which may lead to increased vulnerability to the dictionary attacks. Because the SSO allows a user to (Editor's note: provide more general text – not just the passwords) have one password for accessing multiple applications, it makes it easier for the service providers to enforce more strict rules for the passwords. This helps to improve the network security.

On the other hand, if the user credentials are compromised the impact on the SSO-enabled networks could be greater than on the systems that do not support SSO. To that end it is essential for the SSO to employ secure mechanisms. This section provides an overview of several mechanisms that can be used for supporting SSO.

Editor's note: The pros and cons for selecting GBA for SSO and guidelines for its use will be provided later.

Editor's note: If clause on GBA is removed this one should be removed also.

### 6.11.1 SAML-based mechanism

## 6.12 Single Sign-off

## 7 Security

### Appendix I: IdM Profiles for NGN

[EdNote: *This section describes relationships between ITU-T NGN environment and necessary NGN IdM models to IdM models developed in other Standards or fora (e.g., Liberty Alliance, OASIS, OpenID) (mapping or profile). Contributions invited.*]

### Appendix II: Bibliography

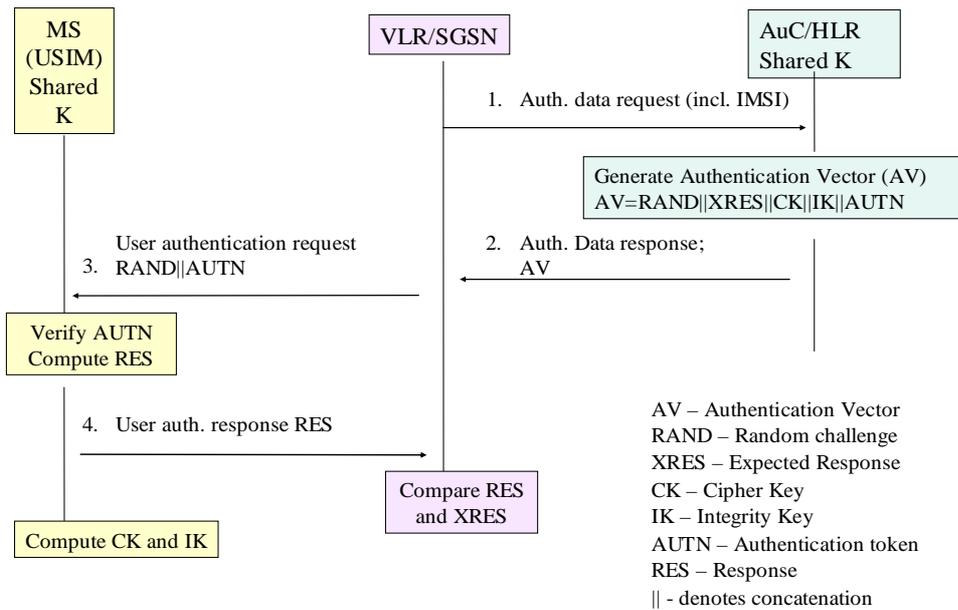
[b-GBA] ETSI TS 133 220, 03/2006, Generic Authentication Architecture (GAA); Generic bootstrapping architecture.

[SAML token] OASIS Standard *Web Services Security: SAML Token Profile 1.1*, (2006)

[XML signature] W3C Recommendation *XML Signature Syntax and Processing*, (2002).

WSS: *Shin will provide the exact information!*

### Appendix III.



**Figure 1 – AKA Message Flow**

1. The VLR/SGSN sends request for authentication data to the subscriber's home authentication center. The request contains the subscriber's IMSI identifier.
2. The AuC/HLR generates an authentication vector composed of a concatenation of the following values: random challenge RAND, expected response XRES, cipher key CK, integrity key IK and authentication token AUTN. All these values are computed with the use of the shared secret K. Then the AuC/HLR sends the authentication vector AV to the VLR/SGSN.
3. The VLR/SGSN sends authentication request to the MS, which includes the value RAND||AUTN.
4. The mobile station's USIM application computes the authentication token itself using the shared secret K and the received value RAND, and compares it to the value of the authentication token AUTN received in step 3. If they match, the USIM proceeds with calculating a response RES and sending it to the VLR/SGSN. It also computes the cipher key CK and integrity key IK.

Finally, VLR/SGSN can authenticate the MS by comparing the response RES (received from the MS) and the expected response XRES (received from the AuC/HLR). If they match then the AKA procedure has resulted in mutual authentication of the MS and VLR/SGSN, which have agreed on the cipher key CK and integrity key IK for securing their further communications.

#### **Appendix IV: Example identities in NGN**

Note: the material here are moved from draft NGN IdM framework (output of May 2008 meeting), considering that this can be a good starting point to develop a text on NGN identities.

## **8 NGN Identities {It is proposed that the information in this section be consolidated, deleted or included in an Appendix}**

[*EdNote*: we must clean up this section

]

[*Editors' Note*: 3GPP has different definitions on Identity and Identifiers. We have to be careful to use ITU-T terms.]

In public NGN infrastructures, large numbers of diverse entities have the ability to autonomously access and control highly-distributed, shared, global ICT resources, as well as communicate with other entities. These entities may include real persons, legal persons (e.g., organizations), and a vast array of objects such as devices, network elements, RFIDs, sensors, and software-based agents.

The entities may be acting as either subscribers or as providers, in public or private capacities. In order to use the public NGN network infrastructure, these entities assert identity claims and are provided various identities that are manifested and used to support NGN Identity Management functions across multiple service/ network provider boundaries within a NGN Identity Management Framework, as described in this Recommendation. An identity consists of the identifiers and other attributes by which an entity is described, recognized or known. Section 9, below, describes capabilities associated with the creation and use of identifiers and related identity attributes that are necessary for integrity and security, integrity, and privacy of NGN infrastructure.

NGN identities in this NGN security framework recommendation apply to subscribers, network/service providers, and objects.

*EdNote*: the text above must be checked whether it aligns with the current definition of identities in 3GPP below.

TS 184 002 describes a concept on the use of identifiers in the NGN. This concept re-uses the notion of private identifiers and public identifiers as defined by TS 123 003. This Recommendation applies this notion of private and public identifiers.

### **8.1 Subscriber Identifiers**

[*EdNote*: this section describes identifiers, not identities. Justification necessary.]

Subscriber entities may assert or claim either public or private identifiers. The term “public” in this context includes identifiers that are generally available to other network subscribers. “Private” means identifiers that is used for authentication purposes only, and is never displayed even to the subscriber.

#### **8.1.1 Public identifiers**

The availability of public identifiers is determined by a complex mix of legal-regulatory-contractual requirements, operational and security needs, and subscriber (so-called *user-centric*) preferences. Public identifiers and any associated policies should be discoverable, authoritative, and obtainable using well-known interoperable, extensible protocols and syntax structures. Public identifiers must be sufficient to permit the communication, action, or transaction to occur.

#### **8.1.2 Private identifiers**

Private Subscriber identifiers are those other than public identifiers, and may include anonymity to the extent permitted by the context of the communication, action, or transaction, and related legal-regulatory-contractual requirements of the relevant jurisdictions.

## 8.2 Network/Service Provider identifiers

Network/Service Provider entities may assert either public or private identifiers. The term “public” in this context includes identifiers that are generally available to both network subscribers and other providers. “Private” means identifiers that is used for authentication purposes only, and is never displayed even to the provider.*EdNote: the text above must be checked whether it aligns with the current definition of identities in 3GPP and other SDO’s definitions below.*

- IP address (including address realm), FQDN.
- Home domain name; see TS 184 002 clause 6.2.1.1.
- SIP URI for NGN network nodes.
- Public service identifier, see TS 184 002 clause 6.2.4.
- Access network identifier, see TS 184 002 clause 6.2.2.
- DNS names.
- CNGCF Address, see TS 184 002 clause 6.2.2.
- P CSCF Identity, see TS 184 002 clause 6.2.2.
- AF Identity, see TS 184 002 clause 6.2.2.
- Resource Reservation Session ID, see TS 184 002, clause A.2.3.
- Charging Correlation identifier, see TS 184 002 clause A.2.3.
- Subscriber Info, see TS 184 002 clause A.2.3.
- Resource Bundle-Id information, see TS 184 002 clause A.2.3.
- RACF identification, see TS 184 002 clause A.2.3.  
*EdNote: this identifier could not be verified; seems not to exist in the referenced specification.*
- Media (session) identifier; see TS 184 002 clause A.2.3.
- Flow identifier; see TS 184 002 clause A.2.3.
- Application Class ID, see TS 184 002 clause A.2.1.

### 8.2.1 Public network/service providers

Network/service providers may assert only public identifiers, and must meet the requirements of 5.1.1, above, as applicable.

### 8.2.2 Private/home network service providers

Private/home network service identifiers are those other than public identifiers, and may include private attributes to the extent permitted by the context of the communication, action, or transaction, and related legal-regulatory-contractual requirements of the relevant jurisdictions.

[**Editorial** note: the category of “private/home network” providers seemed necessary to account for these kinds of entities.]

## 8.3 Object Identities

[*EdNote: current text includes device identities. Contributions are invited to clarify this.*]

NGNs will consist of broad arrays of object entities – both physical and virtual - that connect to or are part of the network infrastructure and assert either public or private identities.

### 8.3.1 Terminal or Sensor Devices

Terminal or sensor devices - including radio-based devices and RFID or similar near-field communication object reader systems - connected to public NGN infrastructures may assert public or private identities, and must meet the requirements of 5.1, above, as applicable.

### 8.3.2 Network-Based Equipment

Network elements that are part of public NGN infrastructures may assert public or private identities, and must meet the requirements of 5.1.1, above, as applicable.

*[EdNote: the text above must be checked whether it aligns with the current definition of identities in 3GPP below.]*

Line identifier.

Physical access ID (including Location Information); see TS 184 002 clause A.2.1.

Logical access ID, including Access Network Type and derived RACS Point of Contact; see TS 184 002 clause A.2.1.

### 8.3.3 Other Objects

Diverse kinds of other physical and virtual objects including agents may assert public or private identities, and must meet the requirements of 5.1.1, above as applicable.

## Appendix V: X.509 v3 Message Authentication

The following example demonstrates a way to process messages with WSS X.509 token, as described in the section 6.6.3.2

```
<?xml version="1.0" encoding="UTF-8"?>
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:sb="urn:liberty:sb:2006-08"
  xmlns:pp="urn:liberty:id-sis-pp:2003-08"
  xmlns:sec="urn:liberty:security:20 06-08"
  xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd"
  xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
  xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <s:Header>
  <!-- see Liberty SOAP Binding Specification for which headers are required and optional -->
  <wsa:MessageID wsu:Id="mid">...</wsa:MessageID>
  <wsa:To wsu:Id="to">...</wsa:To>
  <wsa:Action wsu:Id="action">...</wsa:Action>
  <wsse:Security mustUnderstand="1">
    <wsu:Timestamp wsu:Id="ts">
      <wsu:Created>2005-06-17T04:49:17Z</ wsu:Created >
    </wsu:Timestamp>
    <wsse:BinarySecurityToken
      ValueType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-x509-token-profile-1.0#X509v3 ">
```

```
wsu:Id="X509Token"  
EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-  
security-1.0#Base64Binary">  
MIIB9zCCAWSgAwIBAgIQ...  
</wsse:BinarySecurityToken>
```

```
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">  
  <ds:SignedInfo>
```

binding --> <!-- in general include a ds:Reference for each wsa: header added according to SOAP

```
<!-- include the MessageID in the signature -->  
<ds:Reference URI="#mid">...</ds:Reference>
```

```
<!-- include the To in the signature -->  
<ds:Reference URI="#to">...</ds:Reference>
```

```
<!-- include the Action in the signature -->  
<ds:Reference URI="#action">...</ds:Reference>
```

```
<!-- include the Timestamp in the signature -->  
<ds:Reference URI="#ts">...</ds:Reference>
```

```
<!-- bind the security token (thwart cert substitution attacks) -->  
<ds:Reference URI="#X509Token">  
  <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>  
  <ds:DigestValue>Ru4cAfeBABE...</ds:DigestValue>  
</ds:Reference>
```

```
<!-- bind the body of the message -->  
<ds:Reference URI="#MsgBody">  
  <ds:DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"/>  
  <ds:DigestValue>YgGfS0pi56pu...</ds:DigestValue>  
</ds:Reference>
```

```
</ds:SignedInfo>
```

```
<ds:KeyInfo>
```

```
  <wsse:SecurityTokenReference>  
    <wsse:Reference URI="#X509Token" />  
  </wsse:SecurityTokenReference>
```

```
</ds:KeyInfo>
```

```
<ds:SignatureValue>
```

```
HJJWbvqW9E84vJVQkjLLA6nNvBX7mY00TZhwBdFNDElgscS XZ5Ekw==
```

```
</ds:SignatureValue>
```

```
</ds:Signature>
```

```
</wsse:Security>
```

```
</s:Header>
```

```
<s:Body wsu:Id="MsgBody">
```

```
  <pp:Modify>
```

```
    <!-- this is an ID-SIS-PP Modify message -->
```

```
  </pp:Modify>
```

```
</s:Body>
```

```
</s:Envelope>
```

