International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

Series Y

Supplement 12

(04/2010)

SERIES Y: GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS AND NEXT-GENERATION NETWORKS

ITU-T Y.2720 – Supplement on NGN identity management mechanisms

ITU-T Y-series Recommendations – Supplement 12

ITU-T Y-SERIES RECOMMENDATIONS

GLOBAL INFORMATION INFRASTRUCTURE, INTERNET PROTOCOL ASPECTS AND NEXT-GENERATION NETWORKS

GLOBAL INFORMATION INFRASTRUCTURE	
General	Y.100–Y.199
Services, applications and middleware	Y.200–Y.299
Network aspects	Y.300–Y.399
Interfaces and protocols	Y.400–Y.499
Numbering, addressing and naming	Y.500–Y.599
Operation, administration and maintenance	Y.600–Y.699
Security	Y.700–Y.799
Performances	Y.800–Y.899
INTERNET PROTOCOL ASPECTS	
General	Y.1000–Y.1099
Services and applications	Y.1100–Y.1199
Architecture, access, network capabilities and resource management	Y.1200–Y.1299
Transport	Y.1300–Y.1399
Interworking	Y.1400–Y.1499
Quality of service and network performance	Y.1500–Y.1599
Signalling	Y.1600–Y.1699
Operation, administration and maintenance	Y.1700–Y.1799
Charging	Y.1800–Y.1899
IPTV over NGN	Y.1900–Y.1999
NEXT GENERATION NETWORKS	
Frameworks and functional architecture models	Y.2000–Y.2099
Quality of Service and performance	Y.2100–Y.2199
Service aspects: Service capabilities and service architecture	Y.2200–Y.2249
Service aspects: Interoperability of services and networks in NGN	Y.2250–Y.2299
Numbering, naming and addressing	Y.2300–Y.2399
Network management	Y.2400–Y.2499
Network control architectures and protocols	Y.2500–Y.2599
Future networks	Y.2600–Y.2699
Security	Y.2700–Y.2799
Generalized mobility	Y.2800–Y.2899
Carrier grade open environment	Y.2900–Y.2999

For further details, please refer to the list of ITU-T Recommendations.

Supplement 12 to ITU-T Y-series Recommendations

ITU-T Y.2720 – Supplement on NGN identity management mechanisms

Summary

Supplement 12 to ITU-T Y-series Recommendations provides a description of some example mechanisms that can be used to meet certain identity management (IdM) requirements and deployment needs of NGN.

History

Edition	Recommendation	Approval	Study Group
1.0	ITU-T Y Suppl. 12	2010-04-30	13

Keywords

Authentication, authorization, identity management, next generation network, single sign-on.

FOREWORD

The International Telecommunication Union (ITU) is the United Nations specialized agency in the field of telecommunications, information and communication technologies (ICTs). The ITU Telecommunication Standardization Sector (ITU-T) is a permanent organ of ITU. ITU-T is responsible for studying technical, operating and tariff questions and issuing Recommendations on them with a view to standardizing telecommunications on a worldwide basis.

The World Telecommunication Standardization Assembly (WTSA), which meets every four years, establishes the topics for study by the ITU-T study groups which, in turn, produce Recommendations on these topics.

The approval of ITU-T Recommendations is covered by the procedure laid down in WTSA Resolution 1.

In some areas of information technology which fall within ITU-T's purview, the necessary standards are prepared on a collaborative basis with ISO and IEC.

NOTE

In this publication, the expression "Administration" is used for conciseness to indicate both a telecommunication administration and a recognized operating agency.

Compliance with this publication is voluntary. However, the publication may contain certain mandatory provisions (to ensure e.g., interoperability or applicability) and compliance with the publication is achieved when all of these mandatory provisions are met. The words "shall" or some other obligatory language such as "must" and the negative equivalents are used to express requirements. The use of such words does not suggest that compliance with the publication is required of any party.

INTELLECTUAL PROPERTY RIGHTS

ITU draws attention to the possibility that the practice or implementation of this publication may involve the use of a claimed Intellectual Property Right. ITU takes no position concerning the evidence, validity or applicability of claimed Intellectual Property Rights, whether asserted by ITU members or others outside of the publication development process.

As of the date of approval of this publication, ITU had not received notice of intellectual property, protected by patents, which may be required to implement this publication. However, implementers are cautioned that this may not represent the latest information and are therefore strongly urged to consult the TSB patent database at http://www.itu.int/ITU-T/ipr/.

CONTENTS

Supplement 12 to ITU-T Y-series Recommendations

ITU-T Y.2720 – Supplement on NGN identity management mechanisms

1 Scope

This supplement provides a description of some example mechanisms that can be used to meet certain identity management (IdM) requirements and deployment needs of NGN.

NOTE – This supplement provides only a subset of the IdM mechanisms needed for NGN. The purpose of this supplement is to provide the information in a timely manner while the work to develop a Recommendation specifying a more complete set of IdM mechanisms progresses. The objective of this work is to define the IdM mechanism in support of the requirements specified in [ITU-T Y.2721].

2 References

[ITU-T X.509] Recommendation ITU-T X.509 (2008) | ISO/IEC 9594-8:2008, *Information technology – Open systems interconnection – The Directory: Public-key and atribute certificate frame works*.

[ITU-T X.1035] Recommendation ITU-T X.1035 (2007), *Password-authenticated key exchange (PAK) protocol*.

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

[ITU-T Y.2012] Recommendation ITU-T Y.2012 (2006), *Functional requirements and architecture of the NGN release 1*.

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

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

[ITU-T Y.2720] Recommendation ITU-T Y.2720 (2009), *NGN identity management framework*.

[ITU-T Y.2721] Recommendation ITU-T Y.2721 (2010), *NGN identity management requirements and use cases*.

[ATIS 33102] ATIS.3GPP.33.102V710-2007, *3G Security; Security Architecture*.

[IETF RFC 2289] IETF RFC 2289 (1998), *A One-Time Password System*. <http://www.ietf.org/rfc/rfc2289.txt>

[IETF RFC 2616] IETF RFC 2616 (1999), *Hypertext Transfer Protocol – HTTP/1.1*.

[IETF RFC 2617] IETF RFC 2617 (1999), *HTTP Authentication: Basic and Digest Access Authentication*.

[IETF 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>

[IETF 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>

[IETF RFC 4279] IETF RFC 4279 (2005), *Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)*.

[3GPP TR 33.924] 3GPP TR 33.924 Release 9 (2009), *3rd Generation Partnership Project; Identity management and 3GPP security interworking; Identity management and Generic Authentication Architecture (GAA) interworking (Release 9)*.

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

[OASIS WSS X.509 profile] OASIS (2006), *Web Services Security X.509 Certificate Token Profile 1.1*.

[OASIS SAML token] OASIS (2006), *SAML Token Profile 1.1*, *OASIS Approved Erratta 1*.

[OASIS WSS SOAP] OASIS (2004), *Web Services Security (WSS) SOAP Message Security 1.1*.

[LA ID-WSF security] Liberty Alliance Project (2007), *Liberty ID-WSF Security Mechanisms Core Version 2.0-errata Version 1.0*.

[LA SOAP binding] Liberty Alliance Project, Web Services Security (WSS) (2006), *Liberty SOAP Binding Ver-2.0*.

[W3C XML signature] World Wide Web Consortium (W3C) (2008), *XML Signature Syntax and Processing (Second Edition)*.

3 Definitions

This supplement relies on the terms defined in [ITU-T Y.2720].

4 Abbreviations

This supplement uses the following abbreviations and acronyms:

AKA Authentication and Key Agreement

ASP Application Service Provider

AV Authentication Vector

BSF Bootstrapping Server Function

CK Ciphering Key

GBA Generic Bootstrapping Architecture

HSS Home Subscriber System

IdM Identity Management

IdP Identity Provider

ID-WSF Identity Web Services Framework

IK Integrity Key

IMPI IP Multimedia Private User Identity

IMPU IP Multimedia Public User identity

IMS IP Multimedia Subsystem

IMSI International Mobile Subscriber Identity

IPTV Internet Protocol Television

ISIM IMS Subscriber Identity Module

NAF Network Application Function

NGN Next Generation Network

OASIS Organization for the Advancement of Structured Information Standards

OTP One Time Password

PKI Public Key Infrastructure

SAML Security Assertion Markup Language

SIP Session Initiation Protocol

SLF Subscriber Locator Function

SOAP Simple Object Access Protocol

SSO Single Sign-On

UE User Equipment

UICC Universal Integrated Circuit Card

UMTS Universal Mobile Telecommunications System

WSS Web Services Security

XML eXtensible Markup Language

5 Conventions

None.

6 Mechanisms and procedures supporting IdM functions

This clause describes mechanisms and procedures that support IdM functions.

6.1 Lifecycle management

Refer to [ITU-T Y.2720] for information on identity lifecycle management.

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.

The identity provider (IdP) supports authentication methods such as authentication based on web services (WS) security assertion markup language (SAML) profile, certificate-based authentication, or password-based authentication (including one time password (OTP)). The authentication 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.

6.2.1 Authentication based on the WS security SAML profile

This clause describes SAML-based authentication.

6.2.1.1 SAML assertions

The security assertion markup language (SAML) [ITU-T X.1141] specifies a 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 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 simple object access protocol (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 [W3C XML signature] is used to verify the relationship between the SOAP message and the statements of the assertions carried in the message. The *Web Services Security (WSS): SAML Token Profile* [OASIS SAML token] standard describes how:

- SAML assertions (also referred to as SAML tokens) are carried in and referenced from a SOAP message;
- an 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 a 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.

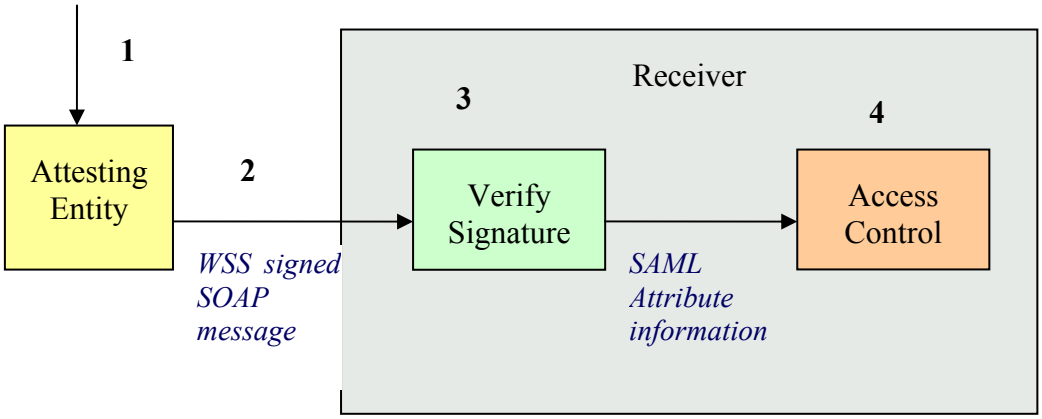

Figure 1 – Typical steps of construction and processing of a SOAP message with a SAML token

1. The attesting entity obtains a SAML assertion with an attribute statement and constructs and includes it in a SOAP message constructed according to [OASIS SAML token].
2. The attesting entity sends a SOAP message to the receiver.
3. The receiver verifies the 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, [OASIS SAML token] specifies how to attach a 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 [OASIS WSS SOAP] are depicted in Figure 2.

The SAML assertion is placed into the <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 and the signature algorithm is RSA_SHA 1 as specified in [OASIS WSS SOAP]. 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.

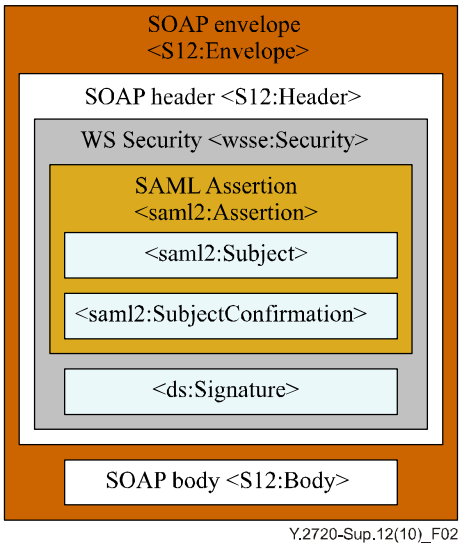

Figure 2 – Structure of the SOAP message with SAML assertion

The following clauses describe two subject confirmation methods.

6.2.1.2.1 The holder-of-key subject confirmation method

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

The method specifies that the sender (also known as the 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 the 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.

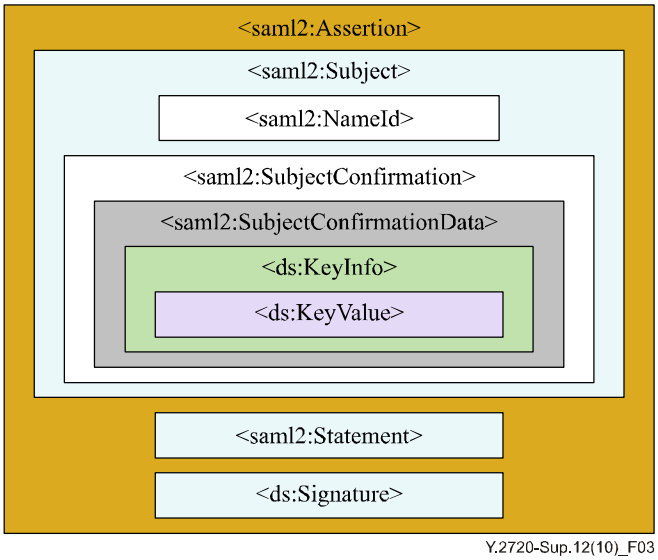

Figure 3 – The structure of the SAML assertion used for the holder-of-key subject confirmation method

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 the digital signature of the SOAP body and checks whether it matches the signature provided by the attesting entity. If it is the case, 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

Figure 4 depicts the structure of the SAML assertion used for the 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 the 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.

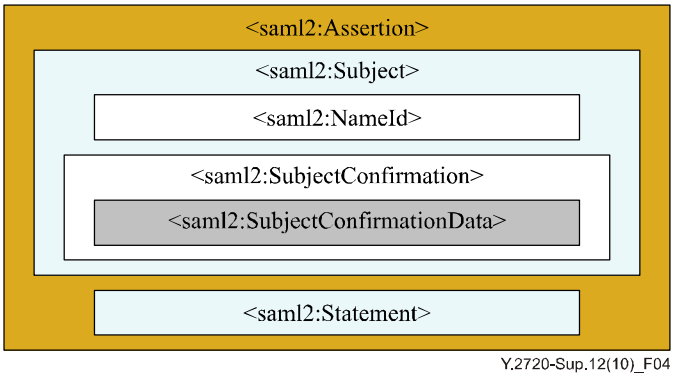

Figure 4 – The structure of the SAML assertion used for the sender-vouches subject confirmation method

6.2.2 Certificate-based authentication

The use of ITU-T X.509 certificates for authentication is described in [ITU-T Y.2704].

6.2.3 Password-based authentication

Depending on the required level of assurance, password-based authentication may be used. Refer to [ITU-T X.1035] for a description of a password-based authentication mechanism.

6.2.4 One-time password

Depending on the required level of assurance, a one-time pawssword (OTP) may be used. One method of implementing the OTP is described in [IETF RFC 2289].

6.2.5 Use of authentication and key agreement (AKA) for mutual authentication

The universal mobile telecommunications system (UMTS) authentication and key agreement (AKA) protocol is used to provide mutual authentication of the mobile station (MS) and the network. The UMTS AKA is a challenge-response protocol, which uses a long-term key shared between the universal subscriber identity module (USIM) and the authentication centre (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 [ATIS 33102].

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. On the other hand, the security of certain NGN services (e.g., IPTV) is based on PKI certificates. To ease the blending of these NGN services with IMS services, it is desirable to integrate PKI-based authentication into the IMS, and to do it in a way that will leverage 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 to be able to perform PKI operations.

With respect to an agreement on the ciphering key (*CK*) and integrity key (*IK*), the described mechanism for integration specifies two options:

1. Establishment of an agreement on the *CK* and *IK* keys with the use of a shared secret between the end-user function and S-5, the service user profile functional entity (SUP-FE).
2. Establishment of an agreement on the *CK* and *IK* keys without the use of a 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 conventions are used in this clause:

"|" 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 Establishment of an 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. The end-user function sends a 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 [ATIS 33102].
3. The S-1 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. The end-user function sends to the S-n a 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.

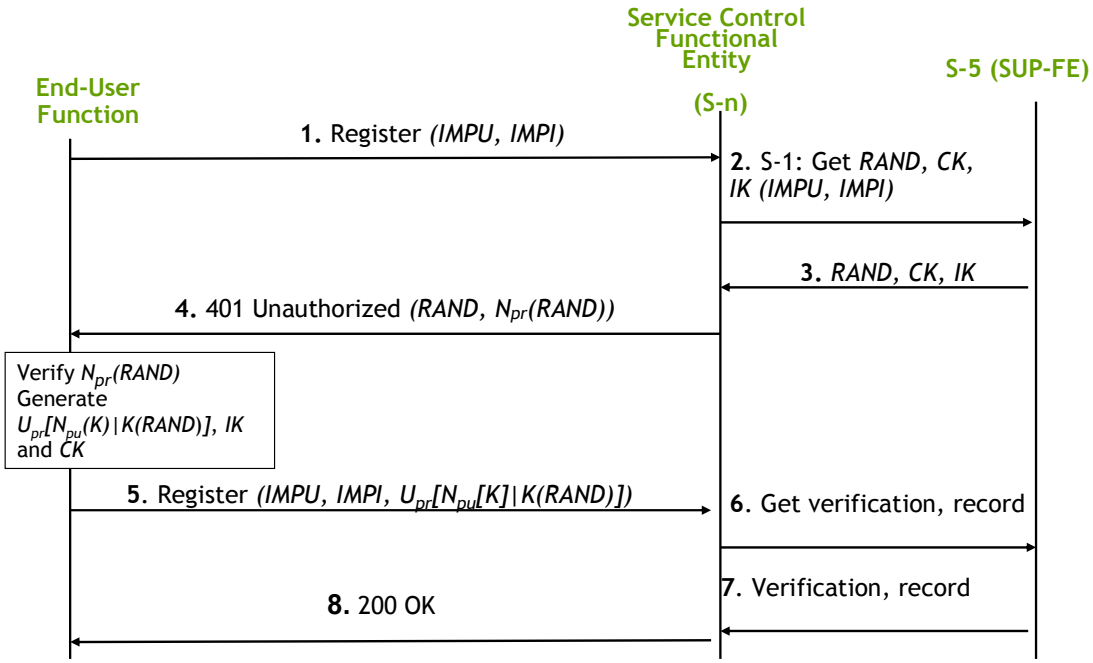

Figure 5 – Integration of the IMS authentication mechanism with PKI-based authentication (option 1)

6.2.6.4 Establishment of an 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. The end-user function sends a 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 [ATIS 33102].
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. The end-user function sends to the S-n an 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 [ATIS 33102] 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.

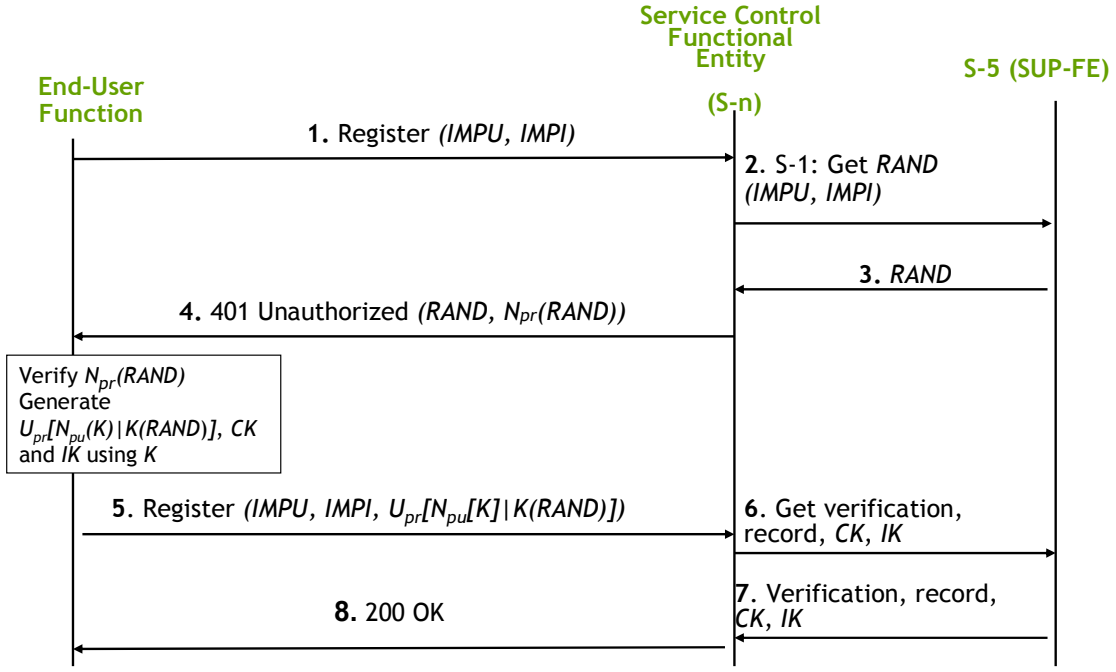

Figure 6 – Integration of the IMS authentication mechanism without PKI-based authentication (option 2)

6.2.6.5 Comparison of options 1 and 2

Table 1 provides a comparison between the mechanisms described for options 1 and 2.

Table 1 – Comparison of options 1 and 2 for 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 an 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 S-5 share a secret. Option 2 should be the choice when the end-user function and S-5 do not have a shared secret.

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

Requirements for the S-5 functional entity

In addition to the capabilities specified in [ATIS 33102], S-5 must be capable of:

- storing the user and network certificates;
- performing PKI-based decryption as described in step 6 (for both options);

- running the Diameter protocol modified to carry the 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 for the end-user function

The end-user function should 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 ITU-T X.509 certificate with the network's public key N_{pu};
- randomly generating the 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 [ATIS 33102] (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 for 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 for 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 Figures 5 and 6.

There are SIP interfaces between:

- the 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 the information described in steps 4 and 5 (for both options).

6.2.6.9 Requirements for 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 the information described in step 6 (for both options).

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

It is desirable that the application servers be 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 (a relying party, such as an application service provider) 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:

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

The IdP has the following benefits:

- it can attract more ASPs by offering its IdM services, particularly authentication;
- the IdP (especially when the 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 [ITU-T 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 [ITU-T 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 [ITU-T 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 [ITU-T X.1141];
- S-5 – Service user profile functional entity (SUP-FE).

The information flow of the authentication procedure is depicted by Figure 7 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 a 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 $K_s(K)$.

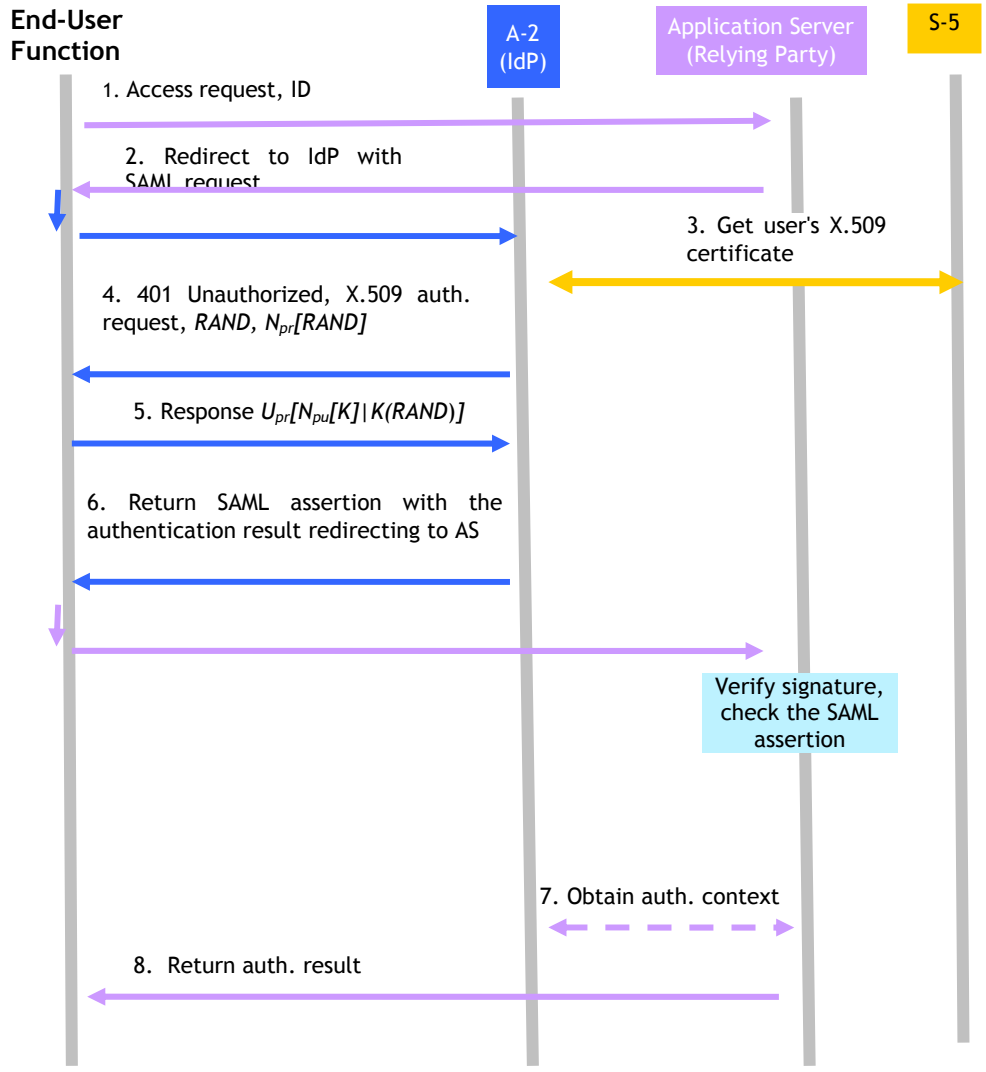

Figure 7 – The basic steps of data exchange for the PKI-based authentication with SAML-assertion

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:

1. 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.
2. 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 [ITU-T 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 [ITU-T X.1141]. The shared secret K_s must be used for signing.
3. 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.
4. A-2 responds to the end-user function with an HTTP response message indicating that authentication with the use of an ITU-T X.509 certificate is required. This is accomplished by setting the value of the response header `www-Authenticate` [IETF 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`.
5. 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*.
6. 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 `pki-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 [ITU-T X.1141]. The shared secret K_s should be used for signing.

  After validating the signed URL, the 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.

7. 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 [ITU-T X.1141].
8. 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 an 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):
  - the AS must support SAML [ITU-T X.1141];
  - the 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 the 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 [ITU-T 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 the HTTP protocol [IETF RFC 2616];
- the interfaces between the end-user function and A-2 functional entities must support the HTTP protocol [IETF RFC 2616];
- the interface between A-2 and the application server must support SAML [ITU-T 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' ITU-T X.509 certificates from S-5.

6.2.8 Integration of *OpenID*-based authentication with IMS

A 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 v.2];
- improves user security by engaging a user-trusted network operator in the access control to the Web applications.

A Technical Report [3GPP TR 33.924] describes several solutions for integration of OpenID with AKA that rely on the use of a bootstrapping server function (BSF).

Another alternative mechanism, described in this clause, allows the integration of OpenID and AKA. To this end, the OpenID specification calls for a variety of authentication mechanisms. This mechanism also supports the split terminal scenario, described in [3GPP TR 33.924].

Entities involved in the authentication and the information flow are as follows:

- 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 v.2] identity provider. (The A-2 optionally shares a short-term secret with the application server as specified in [OpenID v.2]);
- S-5 – Service user profile functional entity (SUP-FE).

The information flow of the authentication procedure is depicted by Figure 8. 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:

Option a – The A-2 and the application server share a secret;

Option b – The A-2 and the application server do not share a secret.

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

Steps 7b, 8b, and 9b are for option b only.

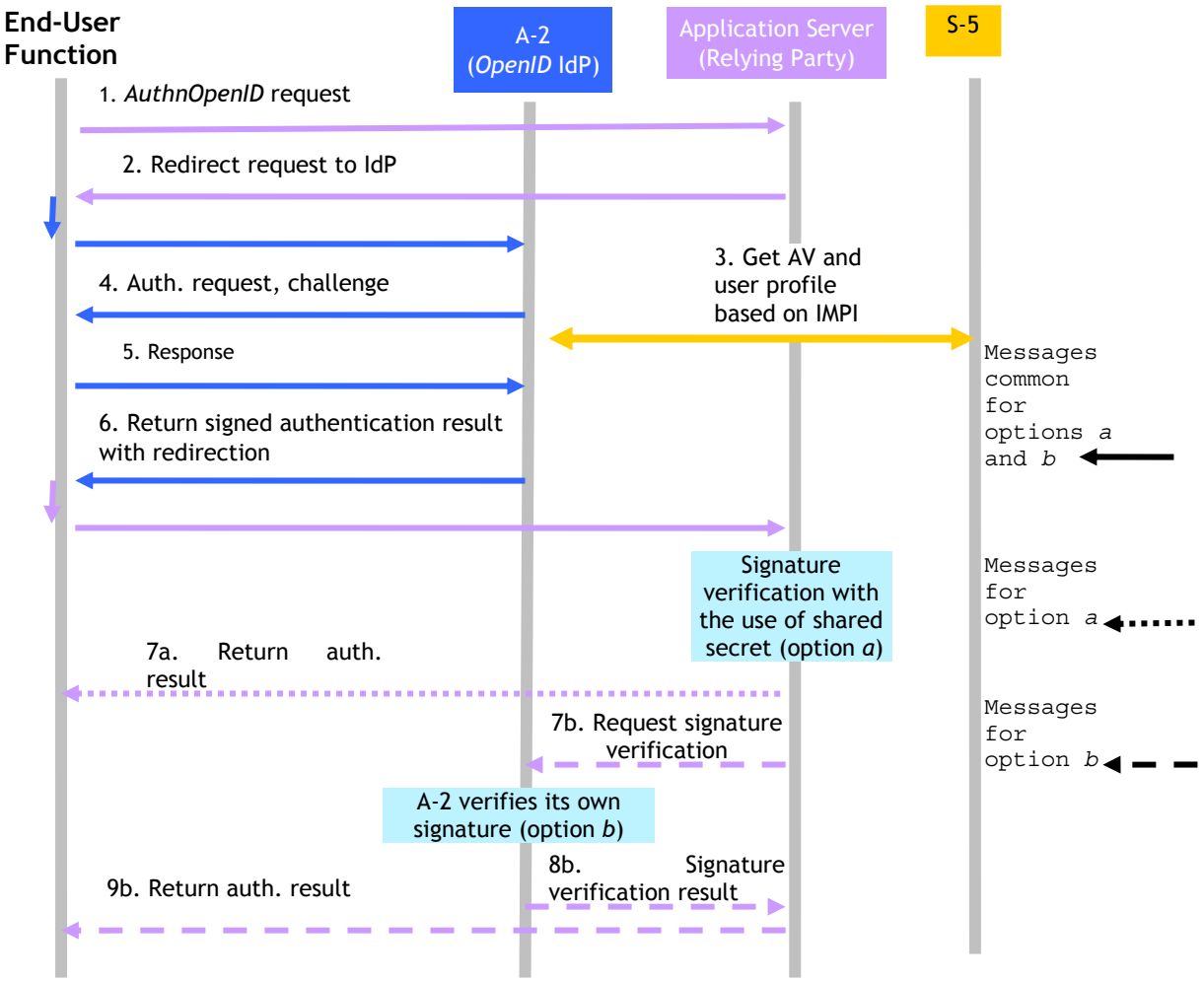

Figure 8 – Integration of the IMS authentication mechanism with OpenID-based authentication

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 S-5 the AKA authentication vector AV and the user profile based on the IMPI.
4. A-2 sends to the end-user function an authentication request using the HTTP Digest AKA method [IETF RFC 4169] or [IETF 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 [IETF RFC 4169] or [IETF RFC 3310].

5. The end-user function sends to A-2 response to the challenge, as specified in [IETF RFC 4169] or [IETF RFC 3310].

   After this step, A-2 authenticates the end-user function, as specified in [IETF RFC 4169] or [IETF RFC 3310].

6. 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 option a. For 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 v.2].

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 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 A-2, with a request to verify the signature.

8b. After verifying its own signature, 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.

6.2.8.1 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 the ISIM application.

- Requirements for the application server:

  The application server must be able to support OpenID specification version 2.0 [OpenID v.2].

- 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 for the S-5 functional entity than those specified in [ITU-T Y.2012].

6.2.8.2 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 [OpenID v.2];
- the interfaces between the end-user function and A-2 functional entities must support the HTTP Digest AKA protocol [IETF RFC 4169] or [IETF RFC 3310];
- the interface between the A-2 and S-5 functional entities does not have any mechanism-specific requirements.

6.2.9 Generic bootstrapping architecture

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 the 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);
- an 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 the two other entities.

The GBA enables authentication of the end-user, who is using the 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 described below (see Figure 9).

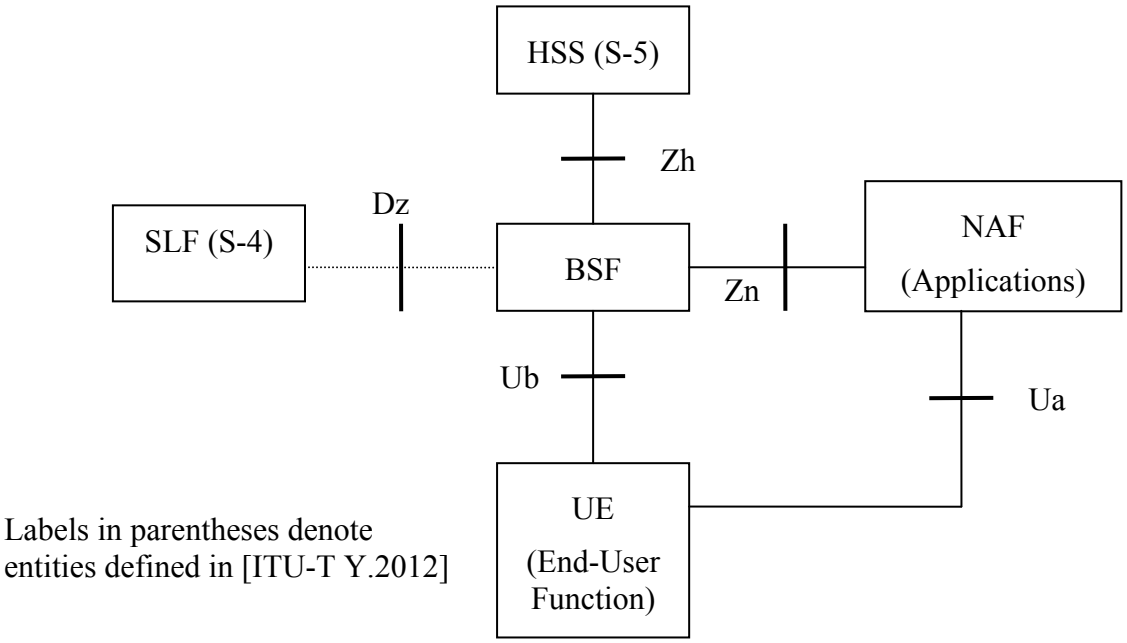

Figure 9 – Simple network model for bootstrapping

These are the basic steps of the GBA procedure:

1. The NAF requests authentication and negotiates the use of GBA over the Ua reference point.
2. The BSF client that runs on the UE initiates the 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 the bootstrapping transaction identifier (B-TID) from the BSF and establishing a shared key (Ks) between the UE and the BSF.
3. The UE derives Ks_NAF from K_s and sends B-TID (along with the application-specific data) to the NAF.
4. The NAF sends B-TID to the BSF over the Zn reference point.
5. The BSF based on B-TID determines the K_s that should be used, derives Ks_NAF from it and sends Ks_NAF to the NAF.
6. Finally, the UE and the NAF can authenticate each other using the shared key Ks_NAF. The exact authentication procedure depends on the protocol between the UE and the NAF. For instance, GBA specifies that HTTP-based applications can use either HTTP Digest authentication [IETF RFC 2617] or TLS pre-shared key ciphersuites [IETF RFC 4279].

NOTE 1 – 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.

The mapping of the GBA entities to the NGN entities specified in [ITU-T Y.2012], is as follows:

- NAF corresponds to the Applications entity of [ITU-T Y.2012], Figure 3 (NGN generalized functional architecture);
- HSS corresponds to S-5 service user profile FE;
- SLF corresponds to S-4 subscription locator FE;

- UE corresponds to the end-user function.

NOTE 2 – No functional entity in the functional architecture defined in [ITU-T Y.2012] corresponds to the BSF.

6.3 IdM communications and information exchange

This clause recommends protocols and mechanisms to communicate and exchange identity information.

6.3.1 Security of IdM communications and exchange

This clause recommends mechanisms to provide integrity and confidentiality protection of IdM communications.

6.3.1.1 Solutions based on SAML 2.0 [ITU-T X.1141]

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. It is required that clause 8.4 of [ITU-T X.1141] be followed when XML signature is used.

For message level confidentiality protection in addition to the secured communication channel, XML Encryption can be used. It is required that clause 8.4 of [ITU-T X.1141] be followed when XML encryption is used.

6.3.1.2 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 messages between 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 protect the packets transmitted via the network connection [LA ID-WSF security].

(1) Transport layer channel protection

In case of using SSL or TLS as a secure network protocol for ID-WSF, it is required that either SSL 3.0, TLS 1.0 or higher be used. 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.

When signing and verifying protocol messages, it is recommended for the communicating entities 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 [OASIS WSS SOAP] to encrypt the SOAP envelope <S:Body> content.

(3) Message integrity rules

The message integrity rules in this clause apply only if [OASIS WSS SOAP] is used for an ID-WSF protocol message bound to SOAP according to [LA SOAP binding].

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 [LA SOAP binding], 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 [OASIS SAML token].

i) Sender processing rules

- The construction and decoration of the <wsse:Security> header element is required to adhere to the rules specified in [OASIS SAML token].
- 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 [OASIS WSS SOAP] and the applicable WSS token profiles (e.g., [OASIS SAML token] 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 [OASIS WSS SOAP] and the appropriate WSS token profiles (e.g., [OASIS SAML token] 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 [W3C XML signature]. 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 ITU-T X.509 token

The semantics and processing rules for mechanisms with a MESSAGE with the value X509 are described in this clause. An example can be found in Appendix I.

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 X.509 message authentication mechanism uses [OASIS WSS X.509 profile] 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 supplement 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, the sending peer is required to perform message authentication by providing proof of possession of the key associated with the ITU-T 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 clause are additional to the generic message authentication processing rules specified in this supplement.

The sender is required to demonstrate possession of the private key associated with the signature generated in conjunction with the WSS X.509 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 [OASIS WSS SOAP]).

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 considers peer entity authentication sufficient for the 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.

6.4 Identity information access control

This clause describes mechanisms for access control for identity information.

6.4.1 SAML-based mechanism for attribute sharing

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

6.5 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 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 simple passwords, which may lead to increased vulnerability to dictionary attacks. Because the SSO allows a user to have one set of authentication credentials for accessing multiple applications, it makes it easier for service providers to enforce more strict rules for establishing the credentials. This helps to improve network security.

On the other hand, if user credentials are compromised, the impact on the SSO-enabled networks could be greater than on systems that do not support SSO. Thus, it is essential for the SSO to employ secure mechanisms.

An example mechanism that can be used for supporting SSO is a GBA-based mechanism.

6.5.1 GBA-based mechanism

Clause 6.2.9 describes the use of the GBA for authenticating a user to any network application function (NAF). Thus, the GBA effectively provides a single mechanism for signing a user to all GBA-enabled NAFs on a network. Indeed, if a user has been signed-on to a NAF, the BSF and UE have already authenticated each other and established a shared key (K_s). Then the procedure of signing on the user to a next NAF will consist of steps 1, 3, 4, 5, and 6 (step 2 is skipped) described in clause 6.2.9. Again, the procedure results in a secret (Ks_NAF) being shared between the UE and a new NAF. This shared secret can be used for authentication between the UE and NAF.

Appendix I

ITU-T X.509 v3 message authentication

The following example demonstrates a way to process messages with a WSS X.509 token, as described in clause 6.3.1.2:

```
<?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-
      wssecurit y-secext-1.0.xsd"
     xmlns:wsu="http://docs.oasis-open.or    g/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.or g/wss/2004/01/oasis-200401-wss-
            x509-token-profile-1.0#X509v3 "
            wsu:Id="X509Token"
            EncodingType="http://docs.oasis-open.org/wss/20 04/01/oasis-200401-
            wss-soap-message-security-1.0#Base64Binar y">
            MIIB9zCCAWSgAwIBAgIQ...
      </wsse:BinarySecurityToken>

      <ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
            <ds:SignedInfo>
                  <!-- in general include a ds:Reference for each wsa: header
                  added according to SOAP binding -->

                  <!-- 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/x
                        mldsig#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>
                  HJJWbvqW9E84vJVQkjjLLA6nNvBX7mY00TZhwBdFNDElgscS 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>
```

SERIES OF ITU-T RECOMMENDATIONS

Series A	Organization of the work of ITU-T
Series D	General tariff principles
Series E	Overall network operation, telephone service, service operation and human factors
Series F	Non-telephone telecommunication services
Series G	Transmission systems and media, digital systems and networks
Series H	Audiovisual and multimedia systems
Series I	Integrated services digital network
Series J	Cable networks and transmission of television, sound programme and other multimedia signals
Series K	Protection against interference
Series L	Construction, installation and protection of cables and other elements of outside plant
Series M	Telecommunication management, including TMN and network maintenance
Series N	Maintenance: international sound programme and television transmission circuits
Series O	Specifications of measuring equipment
Series P	Terminals and subjective and objective assessment methods
Series Q	Switching and signalling
Series R	Telegraph transmission
Series S	Telegraph services terminal equipment
Series T	Terminals for telematic services
Series U	Telegraph switching
Series V	Data communication over the telephone network
Series X	Data networks, open system communications and security
Series Y	**Global information infrastructure, Internet protocol aspects and next-generation networks**
Series Z	Languages and general software aspects for telecommunication systems

Printed in Switzerland
Geneva, 2010