Draft EN 301 261-3 V1.1.1 (1998-04)

European Standard (Telecommunications series)

Telecommunications Management Network (TMN); Security; Part 3: Security services; Authentication of users and entities in a TMN environment

European Telecommunications Standards Institute

Reference
DEN/TMN-00002-3 (bocr0ico.PDF)

Keywords
TMN, security

ETSI Secretariat

Postal address
F-06921 Sophia Antipolis Cedex - FRANCE

Office address
650 Route des Lucioles - Sophia Antipolis
Valbonne - FRANCE
Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16
Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Internet
secretariat@etsi.fr
http://www.etsi.fr
http://www.etsi.org

Contents

Intellectual Property Rights

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

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

Foreword

This European Standard (Telecommunications series) has been produced by ETSI Technical Committee Telecommunications Management Network (TMN), and is now submitted for the Public Enquiry phase of the ETSI standards Two-step Approval Procedure.

The present document is part 3 of a multi-part EN covering security, as identified below:

Part 1: "Framework";

Part 2: "Security support services and security management";

Part 3: "Security services; Authentication of users and entities in a TMN environment";

Part 4: "Security services; Access control".

Proposed national transposition dates	
Date of latest announcement of the present document (doa):	3 months after ETSI publication
Date of latest publication of new National Standard or endorsement of the present document (dop/e):	6 months after doa
Date of withdrawal of any conflicting National Standard (dow):	6 months after doa

Introduction

The authentication service ensures that the identities of the calling and called parties are indeed genuine, that is, they are who they claim they are. Authentication is a first step in establishing secure communications between the calling and called parties.

The verification of an identity can be ascertained only for the instant of the authentication exchange. To guarantee the identity of a communication party for subsequent communication data, the authentication exchange must be used combined with a secure means of communication (e.g. an integrity service).

Authentication is also a support service for many other security services such as e.g. secure exchange of secret keys between calling and called parties.

The authentication service is one of the TMN security services. A complete overview about all TMN security services and the relationships between security services will be given in a framework document.

Prerequisites for the use of the described authentication service (as well as for the use of other TMN security services) are:

- the availability of security support services (like key management etc.);
- the availability of management features for the authentication service; and
- the availability of management features for the security support services.

These prerequisites are described in separate documents.

1 Scope

The present document specifies the authentication service for all kind of users involved in a TMN. Normally, one can distinguish between three types of authentication: (human) user authentication, peer-to-peer entity authentication and data origin authentication. The main scope of the present document is peer-to-peer entity authentication, even if human user authentication is also partly addressed. Data origin authentication will not be addressed as an explicit TMN authentication service for reasons described later in the present document.

The authentication service shall be realized by employing one of a set of various security mechanisms based on password and/or cryptographic means. The main focus of the present document is the description of security mechanisms for peer entity authentication even if these mechanisms may also be applicable for human user authentication. Authentication mechanisms, that may be applicable only for human user authentication, are outside the scope of the present document.

The content of the present document is applicable to communication between any two TMN system entities (e.g. Operations System (OS) and Network Element (NE)) that communicates via a TMN Q3-or an X-interface. The present document addresses peer-to-peer entity authentication at the OSI application layer (layer 7) through the use of ACSE. It does not attempt to cover authentication schemes that may be appropriate for lower OSI layers or other protocol stacks. This does not necessarily restrict the usability of the described authentication services (or part of them) at lower OSI layers or with other protocol stacks.

To the extent that human user authentication is covered, it will be related to the TMN F-interface.

The present document does **not** describe the relationships between authentication service and other security services, the features for managing the authentication service and the authentication support services.

2 References

References may be made to:

a) specific versions of publications (identified by date of publication, edition number, version number, etc.), in which case, subsequent revisions to the referenced document do not apply; or

b) all versions up to and including the identified version (identified by "up to and including" before the version identity); or

c) all versions subsequent to and including the identified version (identified by "onwards" following the version identity); or

d) publications without mention of a specific version, in which case the latest version applies.

A non-specific reference to an ETS shall also be taken to refer to later versions published as an EN with the same number.

2.1 Normative references

[1] ITU-T Recommendation X.509 | ISO 9594-8: "Information Technology - Open Systems Interconnection - The Directory: Authentication Framework".

[2] ITU-T Recommendation X.511 | ISO 9594-3: "Information Technology - Open Systems Interconnection -The Directory: Abstract Service Definition".

[3] ISO/IEC 9798-1: "Information Technology - Security Techniques -Entity Authentication Mechanisms Part 1: General Model, Revision of first edition".

[4] ISO/IEC 9798-2: "Information Technology - Security Techniques -Entity Authentication Mechanisms; Part 2: Entity Authentication using Symmetric Encipherment Algorithms".

[5] ISO/IEC 9798-3: "Information Technology - Security Techniques -Entity Authentication Mechanisms; Part 3: Entity Authentication using a Public Key Algorithm".

[6] ITU-T Recommendation X.227 | ISO/IEC 8650-1: "Data Networks and Open System Communications -Open Systems Interconnection - Connection-mode protocol specifications Information technology - Open Systems Interconnection – Connection-oriented protocol for the association control service element: Protocol specification".

[7] ITU-T Recommendation X.227 AM1 | ISO/IEC 8650 AM1:Series X: "Data Networks and Open System Communications -Open System Interconnection – Connection-mode protocol specifications -Information technology – Open Systems Interconnection – Connection-oriented protocol for the association control service element: Protocol specification - Amendment 1: Incorporation of extensibility markers".

[8] ITU-T Recommendation X.702 | ISO 11587: "Information Technology - Open Systems Interconnection -Application Context for Systems Management with Transaction Processing".

[9] ITU-T Recommendation X.501: "Data Networks and Open System Communications, Directory; Information Technology - Open Systems Interconnection - The Directory: Models".

[10] ISO/IEC 9797: "Information Technology, Security Techniques - Data Integrity Mechanism using a Cryptographic Check Function Employing a Block Cipher Algorithm".

[11] ISO/IEC 10183-3: "Hash Functions - Part 3: Dedicated Hash Functions".

2.2 Informative references

[12] NMF Application Services - Security of Management, Issue 1, 9/92.

[13] IETF RFC1321: "The MD5 Message Digest Algorithm".

[14] FIPSPUB 188: "Standard Security Label for Information Transfer".

[15] FIPS PUB 46-2: "Data Encryption Standard (DES)".

[16] FIPS PUB 186: "Digital Signature Standard (DSA)".

[17] ATM Forum: "ATM Security Specification Version 1.0, STR-SEC-01.01 (Straw Ballot)".

[18] IETF RFC 2078: "Generic Security Service Application Program Interface Version 2".

[19] IETF RFC 2025:"The Simple Public-key GSS-API Mechanism (SPKM)".

[20] IETF RFC 1964: "The Kerberos Version 5 GSS-API Mechanism".

[21] Lai On the design and security of block ciphers, ETH Series in Information Processing, J.L.Massey (editor), vol. 1, Hartung-Gorre Verlag Konstanz, ETH Zurich, 1992.

[22] Public Key Cryptography Standard #1 (PKCS #1): "RSA Encryption Standard, RSA Laboratories, Version 1.5".

[23] ITU-T: "Study group 4, Question 19, STASE-ROSE", Draft, 3/1998.

[24] IETF RFC 2104 HMAC: "Keyed-Hashing for Message Authentication", 2/1997.

3 Definitions and abbreviations

3.1 Definitions

The following terms are defined in ITU-T Recommendation X.509 | ISO 9594-8 [1]:

authentication token, token: see [1].

certificate, user certificate: see [1].

certification authority: see [1].

cryptosystem, cryptographic system: see [1].

hash function: see [1].

one-way function: see [1].

public key: see [1].

private key, secret key: see [1].

simple authentication: see [1].

strong authentication: see [1].

trust: see [1].

The following terms are defined in ISO/IEC 9798-1 [3]:

authentication initiator: see [3].

authentication responder: see [3].

claimant: see [3].

exchange authentication information: see [3].

time variant parameter: see [3].

token: see [3].

trusted third party: see [3].

verification authentication information: see [3].

verifier: see [3].

3.2 Abbreviations

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

ACSE	Association Control Service Element
ATM	Asynchronous Transfer Mode
CMIP	Common Management Information Protocol
DES	Data Encryption Standard
DSA	Digital Signature Algorithm
EN	European Standard (Telecommunications series)
FIPS	Federal Information Processing Standard
FTAM	File Transfer Access Method
GSS-API	Generic Security Service Application Programming Interface
HMAC	Hashed Message Authentication Code
IDEA	International Data Encryption Algorithm
IETF	Internet Engineering Task Force
ISO	International Organization for Standardization
ITU	International Telecommunication Union
MD5	Message Digest Algorithm No. 5
MF	Mediation Function
NE	Network Element
NMF	Network Management Forum
OS	Operations System
OSI	Open Systems Interconnection
PDU	Protocol Data Unit
PKCS	Public Key Cryptography Standard

PUB	Publication
RACE	Research and Development Program for Advanced Communications in Europe
RFC	Request for Comments
ROSE	Remote Operations Service Element
RSA	Rivest, Shamir, Aleman (Algorithm)
SHA	Secure Hash Algorithm
STASE	Security Transformation Application Service Element
TMN	Telecommunications Management Network
WSF	Workstation Function

4 Architectural aspects

The authentication service can be used for intradomain and interdomain TMN (for details see part 1 of " Security for TMN"). The authentication service has two facets:

- generate an authentication token and to transmit it to another communication partner (claimant facet);
- verify an authentication token forwarded by a communication partner (verifier facet).

4.1 General authentication model

As described in [3], the general authentication model involves a calling party, a called party and, if necessary, an authentication party.

If a calling party or a called party interact with an authentication party, then they shall trust the authentication party. The authentication party is a function that can be for example realized as a domain-internal unit (often called Certification Authority) or as part of a TTP (that could be used to settle legal disputes between different domains or legal parties).

The authentication party should only be used for the following tasks:

- generation of (some) authentication information; and/or
- verification of (some) authentication information.

The authentication party should not be used for the (direct) communication of the authentication token between calling and called party.

It is not essential that the authentication party and all the communication exchanges are present in every authentication mechanism. In figure 1, the lines indicate potential information flow. The calling resp. called party may either directly or indirectly interact with the authentication party, or use some information issued by the authentication party.

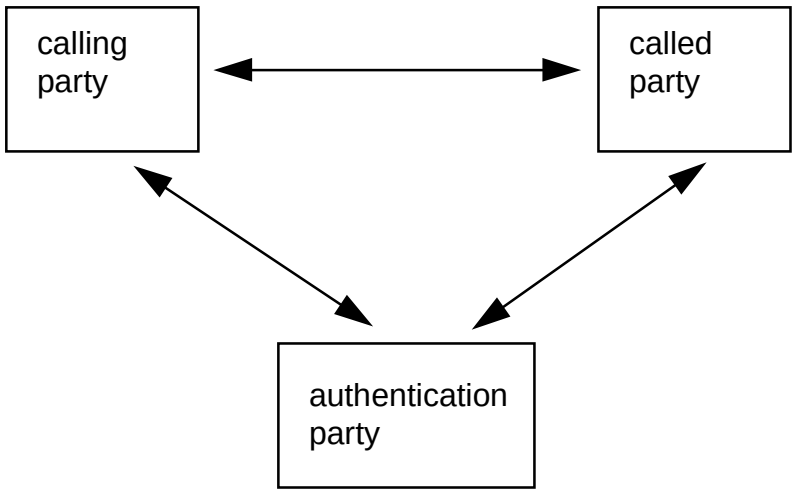

Figure 1: General authentication model

4.2 Mapping the authentication model onto TMN

Dependent on the size of a TMN and on the type of assurance required, different architectures can make sense.

In some cases, it is efficient to realize the authentication service as part of each TMN component. In these cases, no authentication party needs to be involved in authentication. The calling party will generate the authentication information. The authentication information will be transmitted to the called party via the OSI protocol stack. The called party should verify the authentication information and possibly generate new authentication information. This new authentication information will be transmitted from the called party to the calling party as part of the response.

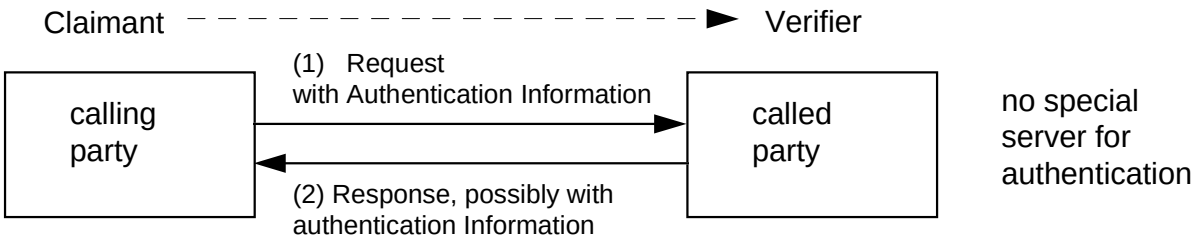

Figure 2: Authentication flow without an authentication party

Currently, the existing Association Control Service Element (ACSE) only allows a single authentication transfer from the calling to the called party and a reply from the called party to the calling party. Therefore, in the present document no authentication mechanism will be considered, which involves more than two information exchanges between calling and called party.

In some TMN environments (e.g. a TMN environment containing many operations systems), it can be advantageous to involve an authentication party (with potential replications in master-slave relationships).

Figure 3 shows an example of such an authentication process:

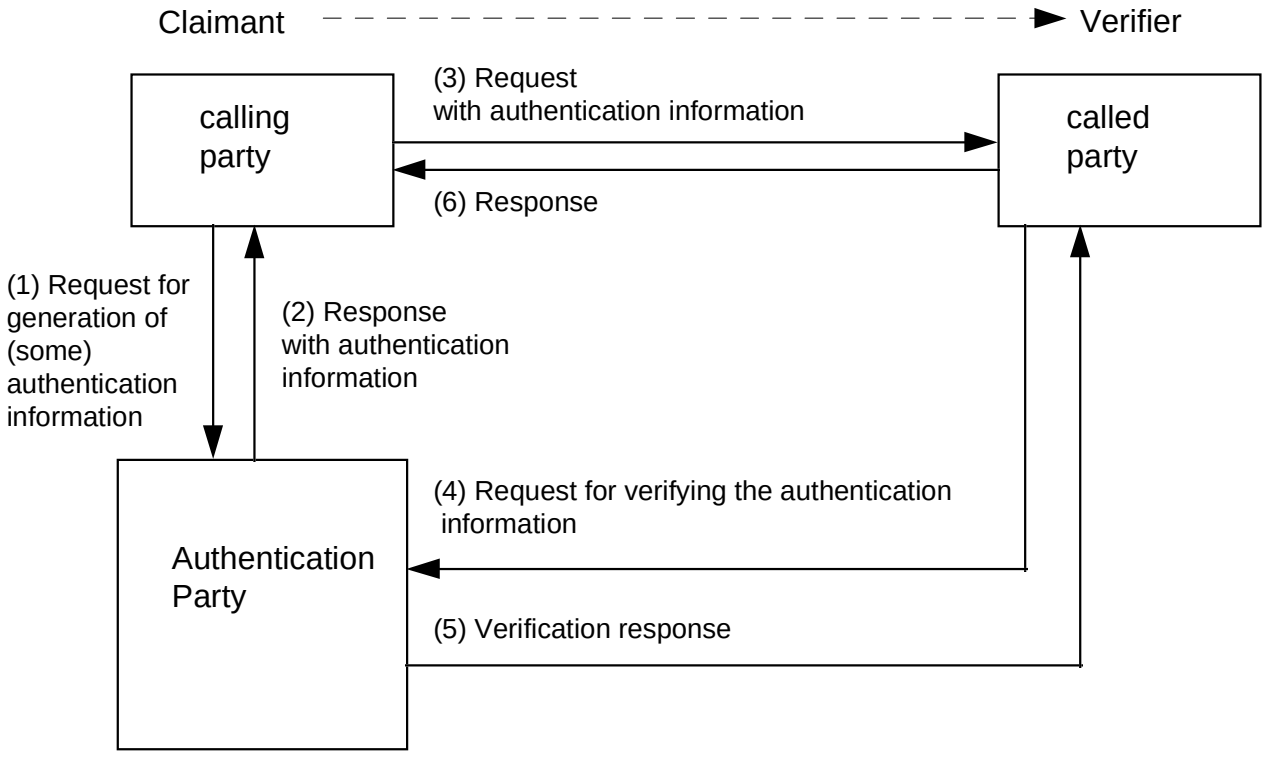

Figure 3: Example of an authentication flow by using an authentication party

The authentication information, which is necessary for an authentication process, will be generated by the authentication party (step 1 and 2). The authentication information will be transmitted via the existing application layer services of the OSI protocol stack (step 3). For verifying the authentication information, the called party may use the authentication party (step 4 and 5). Possibly responses (6) of the called party are: accept or reject.

Currently, the authentication mechanisms described in [4] and [5] only specify the authentication information exchanges between the calling party and the called party. They do not describe (in contradiction to the framework part [3]), how the authentication information will be generated and verified, especially, how an authentication party can be introduced.

In general, an authentication party can be used at each site for the generation and / or for the verification of the authentication information. Therefore, each authentication mechanism leads to the following four implementation variants:

	calling party	**called party**
variant 1	not using an authentication party	not using an authentication party
variant 2	using an authentication party	not using an authentication party
variant 3	not using an authentication party	using an authentication party
variant 4	using an authentication party	using an authentication party

The calling party and the called party may interact with different authentication parties. The data flow and the protocol handling between authentication party and calling resp. called party is outside the scope of the present document. However, it is important to know, that this communication shall be handled in a secure manner.

5 Authentication services

One can distinguish between three types of authentication services: the human user authentication, the peer-to-peer entity authentication and the data origin authentication.

5.1 Human user authentication

Human user authentication provides assurance of the identity of human operators.

The operators are working at a TMN system (i.e. OS, NE, MF) via a Workstation. There exist two scenarios:

- Scenario 1: The Workstation Function (WSF) is physically realized as part of the OS/NE/MF;
- Scenario 2: The WSF is realized in a dedicated WS that is physically separated (connected via an F-interface) from the OS/NE/MF.

In both scenarios, the human user will enter the authentication information to the workstation system (possibly by using a user representation device like chipcard reader).

With the first scenario, the authentication information submitted by the user can be directly verified by the TMN entity that contains the WSF. With the second scenario, there is a question whether the WS is trusted by the OS/NE/MF or not. If the WS is a trusted entity, the authentication verification process may be completed by the WS. If the WS is not trusted, however, the authentication information submitted by the human user must be securely transmitted across the F-interface for verification by the OS/NE/MF.

If a human user is directly involved in the authentication process, then the human user authentication is by definition a unilateral function, i.e. the human user is authenticated towards the workstation system resp. TMN system. Otherwise the human user is represented by a device, which is involved in the authentication process. In this case, the human user is authenticated into two steps: at first he is authenticated towards the user representation device in a unilateral way (e.g. by using a password mechanism) and secondly by applying a peer-to-peer authentication service between the user representation device and the workstation system resp. TMN system.

5.2 Peer-to-peer entity authentication

Peer-to-peer entity authentication provides assurance, during association setup, of the identities of remote TMN entities.

Peer-to-peer entity authentication is a first step in establishing secure communications between systems (e.g. between operations systems and network elements).

Peer-to-peer entity authentication may provide either unilateral and mutual authentication. In the unilateral case, only one of the two communicating systems (the calling party) is authenticated towards the other (the called party), but in the mutual case both communicating systems are authenticated towards each other.

In the unilateral case, either one or two information transfers between calling party and called party are appropriate.

In the mutual case, information will be transferred both from the calling to the called party and from the called to the calling party.

5.3 Data origin authentication

Data origin authentication provides assurance that a data item originates from an identified remote entity.

For data origin authentication, the calling system appends to the data a cryptographic checksum. Data origin authentication is only relevant in a unilateral way.

In a connection-oriented environment like TMN, communicating entities authenticate each other at the time of association establishment. In this case, it should not be required to explicitly " reauthenticate" every message that is transferred on the association by using a data origin authentication service. Nevertheless, the integrity of the established association should be guaranteed. The issue of maintaining the integrity/authenticity of an already established association should be achieved by using an integrity service rather then by using an explicit data origin authentication service (even if the security mechanism used to realize these two different services sometimes can be the same). Therefore, data origin authentication will not be considered as an explicit authentication service for TMN.

6 Authentication mechanisms

An authentication service as described in clause 5 shall be realized by employing one of a set of various security mechanisms. The present document includes mechanisms based on techniques using passwords as well as using cryptographic means.

Password-based authentication is a traditional technique but rather weak and vulnerable. The use of passwords may be improved by using one-way hashing functions applied to time stamps and/or numbers.

From a security point of view, strong authentication, that is use of cryptographic techniques for creation of tokens, are preferred.

A prerequisite to be able to use an authentication mechanism is to have support for management of passwords / keys. Password/key management is outside the scope of the present document.

It shall be possible for two communicating entities to agree out-of-band or to negotiate at the time of association establishment various details regarding the authentication mechanism to use (details for negotiation see clause 8).

The choice of a specific mechanism depends on the security policy and on the TMN system environment to which the mechanism is to be applied.

All of the cryptography based mechanisms described hereafter must be realized through the use of a cryptographic algorithm. A discussion on possible algorithms for the different mechanisms are provided n an informative annex B.

In the following clauses, no mechanisms are described , which make use of more than two communication exchanges (see subclause 4.2). All mechanisms can be used for human user authentication and for peer-to-peer entity authentication. The cryptography based mechanisms assume a user representation device for human user authentication. Human user authentication does not depend on specific characteristics of the ACSE protocol. Therefore, it is also

possible to use human user authentication mechanisms, which are not part of the present document (for example, mechanisms requiring more than two communication exchanges like a challenge - response mechanism).

Most of the mechanisms make optionally use of a time stamp in order to control uniqueness / timeliness. They enable the replay of previously transmitted messages to be detected. The time stamp is controlled by the verifier. Verifier and claimant shall use a common time reference and must have the same understanding of the time stamp. A time stamp can be understood as expiry time of an authentication token or as generation time of an authentication token.

The strength of an authentication procedure depends on several factors such as the mechanisms, the algorithms and/or the passwords. Hereafter, only the strength of a mechanism will be evaluated.

6.1 Authentication mechanisms using passwords

Authentication mechanisms using passwords are based on a clear text or one time password input.

They include different password schemes. Clear text password and one time password transfers are covered by mechanism 1 and 3. Replay protected password transfers are covered by mechanism 2 and 4.Though all schemes are based on passwords, the transferred information and therefore the strength differ.

More details of the mechanisms hereafter are described in [1].

6.1.1 Unilateral authentication

Authentication Mechanism 1: *Simple Unilateral Authentication I based on [1]*

This mechanism uses for authentication a token containing:

optionally	user / system name of the claimant	
optionally	system name of the verifier	
optionally	name of the authentication mechanism	
optionally	name of the algorithm	
mandatory	password	either a clear text password or a one time password

Strength of the mechanism: Low (clear text password) to medium (one time password)

Authentication Mechanism 2: *Simple Unilateral Authentication II based on [1]*

This mechanism uses for authentication a token containing:

optionally	user / system name of the claimant	
optionally	system name of the verifier	
optionally	time stamp	time stamp is mandatory, if a random number is present
optionally	unique number	unique number is either a sequence number or a random number time stamp and/or unique number shall be present
optionally	application specific text field	
optionally	name of authentication mechanism	
optionally	name of the algorithm	
mandatory	replay protected password	the result of a one way function applied to the password and the string above

For replay protection of the password transfer, a one way function shall be applied to the password and the used data fields. This one way function may e.g. be a one way hash function applied once or several times.

Strength of the mechanism: Medium to high

6.1.2 Mutual authentication

Authentication Mechanism 3: *Simple Mutual Authentication I based on [1]*

This mechanism uses for each authentication step a token which has the same token structure as the authentication token of authentication mechanism 1.

Strength of the mechanism: Low (clear text password) to medium (one time password)

Authentication Mechanism 4: *Simple Mutual Authentication II based on [1]*

This mechanism uses for each authentication step a token which has the same token structure as the authentication token of authentication mechanism 2.

Strength of the mechanism: Medium to high

6.2 Authentication mechanisms based on secret keys

Authentication mechanisms based on secret keys makes use of a secret key, known to the involved systems prior to the authentication process. The claimant uses the secret key to generate the authentication token by enciphering a data string. The verifier uses the secret key for verifying the correctness of the authentication token.

The authentication mechanisms hereafter are described in detail in [4]. Each authentication token described hereafter may contain an application specific text field. This field can be used for the transfer of any information, e.g. session keys. The content of this field is outside the scope of the present document.

Each claimant may send some enciphered information generated by an authentication party (e.g. a secret key) together with the authentication token to the verifier.

6.2.1 Unilateral authentication

Authentication Mechanism 5: *Secret Key Based Unilateral Authentication based on [4]*

This mechanism uses for authentication a token containing:

mandatory	enciphered string containing:		
	optionally	name of the claimant	
	optionally	name of the verifier	
	optionally	time stamp	time stamp is mandatory, if a random number is present
	optionally	unique number	unique number is either a sequence number or a random number time stamp and/or unique number shall be present
	optionally	application specific text field	
	optionally	name of the authentication mechanism	
	optionally	name of the algorithm	
	optionally	checksum	to ensure the integrity of the string above the checksum is the result of a one way function applied to the string above
optionally	name of the algorithm		

For the purpose of authentication, it is sufficient to encipher only the checksum (seal). However, the standard [4] requires the encipherment of the whole string. Therefore, the present document only retains this option.

Strength of the mechanism: High

6.2.2 Mutual authentication

Authentication Mechanism 6: *Secret Key Based Mutual Authentication based on [4]*

This mechanism uses for each authentication step a token which has the same token structure as the authentication token of authentication mechanism 5.

Strength of the mechanism: High

6.3 Authentication mechanisms based on public keys

Public key authentication involves a pair of keys, one private and one public, instead of a single shared secret key as in secret key based authentication. The private key is preferably known to the owner only, the complementary public key may be known to many systems prior to the commencement of the authentication mechanism.

For authentication of the claimant system (or a connected authentication party) shall sign an authentication string by using claimant system its private key. To verify the correctness of this authentication string, the verifier system (or a connected authentication party) shall verify the signature by using the complementary public key of the claimant system.

The authentication mechanisms below are described in detail in [5]. Each authentication token described below may contain an application specific text field. This field can be used for the transfer of any information, e.g. secret keys. The content of this field is outside the scope of the present document.

Each claimant may send its public key certificate together with the authentication token to the verifier.

6.3.1 Unilateral authentication

Authentication Mechanism 7: *Public Key Unilateral Authentication based on [5]*

This mechanism uses for authentication a token containing:

optionally	name of the claimant	
optionally	name of the verifier	
optionally	time stamp	time stamp is mandatory, if a random number is present
optionally	unique number	unique number is either a sequence number or a random number time stamp and/or unique number shall be present
optionally	application specific text field	
optionally	name of the authentication mechanism	
optionally	name of the algorithm	
mandatory	signature of the string above	

Strength of the mechanism: High

6.3.2 Mutual authentication

Authentication Mechanism 8: *Public Key Mutual Authentication based on [5]*

This mechanism uses for each authentication step a token which has the same token structure as the authentication token of authentication mechanism 7.

Strength of the mechanism: High

7 Communication protocol mapping

7.1 Human user authentication

With scenario 1 (described in subclause 5.1) human user authentication will directly be done at the workstation. If human users are locally working at the workstation, there will be no transfer of authentication information over an open interface. Otherwise there is a need to transfer authentication information over an open interface. This transfer is outside the scope of the present document.

With scenario 2, there might be a need to transfer authentication information over the F-interface between the Workstation and the connected TMN system.

Since the communication protocols for the F-interface are not standardized, it is left outside the scope of the present document to specify any particular protocol mapping for the transfer of authentication information at this interface.

7.2 Peer-to-peer entity authentication

Input is given by [12].

7.2.1 General Procedure

Peer-to-peer entity authentication shall be part of the association establishment.

The authentication information will be transmitted via ACSE:

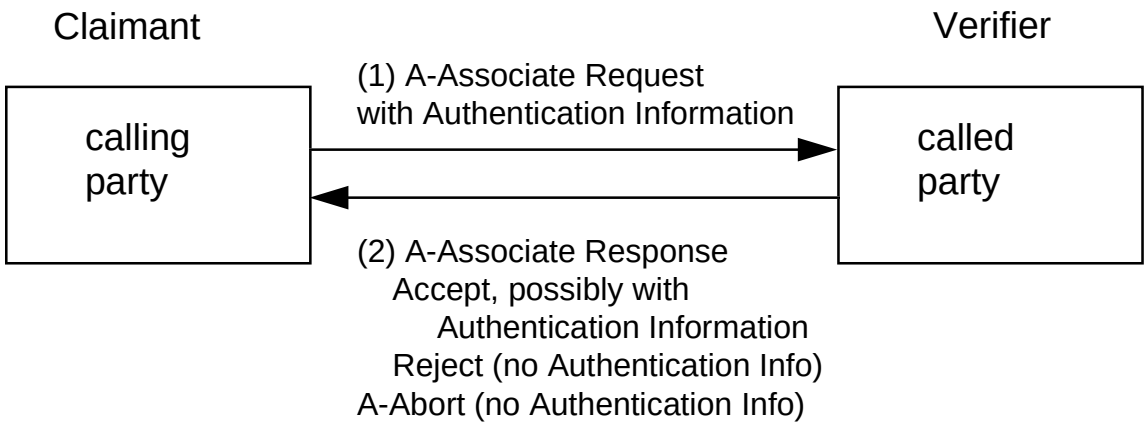

Figure 4: Mapping Authentication Information into ACSE

When sending an A-ASSOCIATE request, the association initiator shall use the ACSE authentication functional unit in the A-ASSOCIATE PDU. The sender-acse-requirements, mechanism-name and calling-authentication-value fields, as defined in the ACSE protocol syntax, shall be present and be used in the following way:

- the sender-acse-requirements shall include the authentication functional unit;
- the mechanism-name shall identify authentication-mechanism and;

- the calling-authentication-value shall contain an authentication token as specified in the next clause.

Upon receiving the association request, the association responder verifies the authentication information and determines whether the association request is to be accepted. The association responder then sends one of the following messages to the association initiator:

- an A-ASSOCIATE response indicating that it has accepted the association;
- an A-ASSOCIATE response indicating that it has rejected the association or;
- an A-ABORT indicating that it has aborted the association.

If the association is accepted, then the responder shall distinguish between unilateral authentication and mutual authentication in the following way:

In the case of unilateral authentication, the sender-acse-requirements shall be present in the A-ASSOCIATE response APDU. The sender-acse-requirement parameter shall include the authentication functional unit. The mechanism-name and calling-authentication-value parameters shall not be present.

In the case of mutual authentication, the responder shall fill in the A-ASSOCIATE response APDU with authentication information in the same manner as in the A-ASSOCIATE request:

The response-acse-requirements, mechanism-name and responding-authentication-value fields shall be present and be used in the following way:

- the response-acse-requirements shall include the authentication functional unit;
- the mechanism-name shall identify authentication-mechanism and;
- the responding-authentication-value shall contain an authentication token as specified in the next clause.

If the verification of the responding authentication token fails, the association initiator then sends to the association responder an A-ABORT message, indicating that it has aborted the association.

Authentication information will be transmitted independent of the used management protocol (CMIP, FTAM).

7.2.2 Specification of the authentication information syntax in ACSE

For transfer of authentication information the ACSE functional unit for authentication shall be used. The ACSE functional unit for authentication is described in [7] as:

```
ACSE-1 {joint-iso-ccitt association-control(2) modules(0) apdux(0) version(1) }
DEFINITIONS ::=
BEGIN
ACSE-requirements ::= BIT STRING { authentication(0) }
Mechanism-name ::= OBJECT IDENTIFIER
-- The mechanism name shall be present, if Authentication-value is of type " other"  .
Authentication-value    ::=     CHOICE {
                        charstring      [0]         IMPLICIT GraphicString,
                        bitstring       [1]         IMPLICIT BIT STRING,
                        external        [2]         IMPLICIT EXTERNAL,
                        other           [3]         IMPLICIT SEQUENCE {
                    other-mechanism-name  MECHANISM-NAME.&id ({ObjectSet}),
                    other-mechanism-value MECHANISM-NAME.&Type ({ObjectSet}{@.other-mechanism-name})
                                                        }
      }
--  The abstract syntax of (calling/responding) authentication-value is determined by the -----
--  authentication mechanism used during association establishment.  The authentication
--  mechanism is either explicitly denoted by the &id field (of type OBJECT IDENTIFIER) for a --
--  mechanism belonging to the class  MECHANISM-NAME, or it is known implicitly by
--  prior agreement between the communicating partners.  If the " other"  component is chosen, then
--  the " mechanism-name"  component must be present in accordance with
--  ITU-T Rec. X.680 | ISO/IEC 8824. If the value " mechanism-name"  occurs in the AARQ-apdu or
--  the AARE-apdu, then that value must be the same as the value for " other-mechanism-name"

MECHANISM-NAME ::=TYPE-IDENTIFIER
ObjectSet MECHANISM-NAME ::= {...}

END
```

ObjectSet MECHANISM-NAME shall be used for the definition of possible constraints between the used authentication token and the used authentication mechanism. The following constraint shall applied to the transfer of the authentication information:

"AuthenticationToken IDENTIFIED BY Authentication-mechanism"

The authentication information shall be defined as:

```
Authentication-Information {new ETSI object identifier} DEFINITIONS ::=
BEGIN
IMPORTS
-- The information type DistinguishedName is defined in [9]. The information types
-- Certification, CertificationPath and AlgorithmIdentifier and the macros SIGNATURE and
-- ENCRYPTED are defined in [1]. The information type AE-title is defined in [6]. These types resp.
-- macros shall be imported:
    DistinguishedName
                FROM InformationFramework { joint-iso-ccitt(2) ds(5) module(1)
                                                              informationFramework(1) },
    CertificationPath, AlgorithmIdentifier, SIGNATURE{ }, ENCRYPTED{ }
                FROM AuthenticationFramework { joint-iso-ccitt ds(5) module(1)
                                                                  authenticationFramework (7) }
    AE-title
                FROM ACSE-1 {joint-iso-ccitt association-control(2) abstract-syntax(1) apdus(0)
                                        version(1)}
-- The " other"  construct of the ACSE authentication value shall be used, if both communication
-- sites support the ASN.1 syntax defined in 1994.
-- The " external"  construct of the ACSE authentication value shall be used, if one of the sites
will
-- not support the ASN.1 syntax defined in 1994.
Authentication-mechanism OBJECT IDENTIFIER ::= { new ETSI object identifier }
-- This object identifier is only relevant, if the " other"  construct of the ACSE authentication
value
-- will be used

Authentication-value        ::=     CHOICE {
                    explicit                    [0]     ExplicitAuthenticator,
                    -- to be used only in [23], not relevant for the present document
                    gssApiAuthenticator [1]     GssApiAuthenticator,
                    general                 [2]     AuthenticationToken
                                                }

GssApiAuthenticator ::=     SEQUENCE {
                    gssMechanism                [0]     OBJECT IDENTIFIER,
                    gssInitialContextToken      [1]     OCTET STRING
                                                    }

AuthenticationToken ::=     CHOICE {
                    simple                  [0]     SimpleToken,
                    secretKey               [1]     SecretKeyToken,
                    publicKey               [2]     PublicKeyToken
                                }

SimpleToken             ::=     SEQUENCE {
        senderName                  [0]     Name                    OPTIONAL,
            -- name of the claimant
        receiverName                    [1]     Name                    OPTIONAL,
            -- name of the verifier
        time                            [2]     GeneralizedTime     OPTIONAL,
            -- time stamp is mandatory, if a random number is present
            -- only, if mechanism 2 and 4 will be used
        uniqueNumber                [3]     BIT STRING          OPTIONAL,
            -- unique number is either a sequence number or a random number
            -- time and/or unique number shall be present
            -- only, if mechanism 2 and 4 will be used
        applicationString               [4]      ApplString             OPTIONAL,
            -- this string is specific to an application
        authMechanName          [5]     MechanName          OPTIONAL,
            -- name of the used authentication mechanism, only if the context is not clear
        authAlgorithmName           [6]     AlgorithmName       OPTIONAL,
            -- name of the used algorithm, only if the context is not clear
        password                        [7]     OCTET STRING
            -- password means not protected password, one time password or replay protected
            -- password (i.e. a one-way hashing function will be applied to the password and the
            -- parameters above)
                                                    }

SecretKeyToken  ::=     SET {
            authPartyString         [0]     SecretAuthPartyString       OPTIONAL,
            authenticationString        [1]     SecretAuthenticationToken,
            authAlgorithmName       [2]     AlgorithmName               OPTIONAL
                -- name of the used algorithm, only if the context is not clear}
```

```
SecretAuthPartyString   ::=     ENCRYPTED { CHOICE {
                            charstring      [0]         IMPLICIT GraphicString,
                            intstring           [1]         IMPLICIT INTEGER,
                            bitstring           [2]         IMPLICIT BIT STRING,
                            external            [3]         IMPLICIT EXTERNAL,
                    other           [4]         IMPLICIT SEQUENCE {
                            other-authParty-name  AUTHPARTY-NAME.&id ({ObjectSet}),
                            other-authParty-value
                          AUTHPARTY-NAME.&Type ({ObjectSet}{@.other-mechanism-name})
                                                                  }
                                                                  }}

AUTHPARTY-NAME ::= TYPE-IDENTIFIER
ObjectSet AUTHPARTY-NAME ::= {...}
authParty-name  ::= OBJECT IDENTIFIER

SecretAuthenticationToken  ::=  ENCRYPTED { SEQUENCE {
        senderName          [0]     Name                        OPTIONAL,
            -- name of the claimant
        receiverName            [1]     Name                        OPTIONAL,
            -- name of the verifier
        time                        [2]     GeneralizedTime         OPTIONAL,
            -- time stamp is mandatory, if a random number is present
        uniqueNumber        [3]     BIT STRING              OPTIONAL,
            -- unique number is either a sequence number or a random number
            -- time and/or unique number shall be present
        applicationString       [4]      ApplString                 OPTIONAL,
            -- this string is specific to an application
        authMechanName      [5]     MechanName          OPTIONAL,
            -- name of the used authentication mechanisms, only if the context is not clear
        authAlgorithmName       [6]     AlgorithmName       OPTIONAL,
            --name of the used algorithm, only if the context is not clear
        checksum                    [7]     INTEGER             OPTIONAL
                        }}

PublicKeyToken ::=   SEQENCE {
        senderName              [0]     Name                        OPTIONAL,
            -- name of the claimant
        receiverName                [1]     Name                        OPTIONAL,
            -- name of the verifier
        time                            [2]     GeneralizedTime         OPTIONAL,
            -- time stamp is mandatory, if a random number is present
        uniqueNumber            [3]     BIT STRING              OPTIONAL,
            -- unique number is either a sequence number or a random number
            -- time and/or unique number shall be present
        applicationString           [4]      ApplString                 OPTIONAL,
            -- this string is specific to an application
        authMechanName      [5]     MechanName              OPTIONAL,
            -- name of the used authentication mechanisms, only if the context is not clear
        authAlgorithmName       [6]     AlgorithmName           OPTIONAL,
            --name of the used algorithm, only if the context is not clear
        tokenSignature          [7]     SIGNATURE { OCTET STRING },
                -- all parameters above shall be signed by the token signature
        certificate                 [8]     SignatureCertificate        OPTIONAL
                -- the sender's public key certificate for the key used for the signature
                                                            }

Name                ::=     CHOICE {
                                                                distinguishedName       [0]
    DistinguishedName,
                                        charName                    [1]     GraphicString,
                                        applicationName         [2]     AE-title
                                    }

MechanName      ::=     CHOICE {
                                                                distinguishedName   [0]
    DistinguishedName,
                                                identifier              [1]     OBJECT IDENTIFIER}


AlgorithmName   ::=     CHOICE {
                                                                distinguishedName   [0]
    DistinguishedName
                                                identifier                  [1]
    AlgorithmIdentifier }

SignatureCertificate    ::=     CHOICE  {
                                                certificate         [0]     Certificate,
                                                certificationPath   [1]     CertificationPath
                                                }

ApplString          ::=     CHOICE {
```

```
                                    charstring      [0]         IMPLICIT GraphicString,
                                    intstring           [1]         IMPLICIT INTEGER,
                                    bitstring           [2]         IMPLICIT BIT STRING,
                                    external            [3]         IMPLICIT EXTERNAL,
                          other      [4]          IMPLICIT SEQUENCE {
                                 other-appl-struct-name  STRUCTURE-NAME.&id ({ObjectSet}),
                                 other-appl-struct-value   STRUCTURE-NAME.&Type
                                 ({ObjectSet}{@.appl-struct-name})
                }

STRUCTURE-NAME  ::= TYPE-IDENTIFIER
ObjectSet STRUCTURE-NAME    ::= {...}
Appl-struct-name    ::= OBJECT IDENTIFIER

END
```

If the association responder does not accept the request to associate, it may send an ABRT PDU or a response indicating that it has rejected the association . The ABRT PDU and the diagnostic values are defined in [6] as follows:

```
ABRT-apdu ::= [ APPLICATION 4 ] IMPLICIT SEQUENCE
      { abort-source                                    [0]     IMPLICIT ABRT-source,
         abort-diagnostic                               [1]     IMPLICIT ABRT-diagnostic  OPTIONAL,
    --  This field shall not be present if only the Kernel is used.
        ..., ...,
        user-information                                [30]        IMPLICIT Association-information
OPTIONAL
      }

ABRT-diagnostic ::= ENUMERATED
      { no-reason-given (1),
        protocol-error (2),
        authentication-mechanism-name-not-recognized (3),
        authentication-mechanism-name-required (4),
        authentication-failure (5),
        authentication-required (6),
        ...
      }

Associate-source-diagnostic  ::=  CHOICE
      { acse-service-user       [1] INTEGER
                                    {   null (0),
                                        no-reason-given (1),
                                        application-context-name-not-supported (2),
                                        calling-AP-title-not-recognized (3),
                                        calling-AP-invocation-identifier-not-recognized (4),
                                        calling-AE-qualifier-not-recognized (5),
                                        calling-AE-invocation-identifier-not-recognized (6),
                                        called-AP-title-not-recognized (7),
                                        called-AP-invocation-identifier-not-recognized (8),
                                        called-AE-qualifier-not-recognized (9),
                                        called-AE-invocation-identifier-not-recognized (10),
                                        authentication-mechanism-name-not-recognized (11),
                                        authentication-mechanism-name-required (12),
                                        authentication-failure (13),
                                        authentication-required (14)
                                        } (0..14 ,  ...),
        acse-service-provider   [2] INTEGER
                                        {   null (0),
                                            no-reason-given (1),
                                            no-common-acse-version (2)
                                            }(0..2 , ...)
      }
```

Partial protocol implementation conformance statements for ACSE authentication information are described in annex C.

8 Authentication parameter negotiation

The context, which mechanism and algorithm shall be supported for authentication, can be established by using out-of-band negotiation/security management (for details see part 3 of "Security for TMN") prior to the commencement of the association establishment. One example could be that the communicating partners agree upon a set of default parameters to be used.

In addition to the out-of-band negotiation, the communication parties can negotiate at the time of association establishment which mechanism and algorithm they will support for a specific association. Negotiation is done in the following way:

The claimant will optionally include the mechanism name and/or the algorithm name as part of the authentication token. If these parameters are not acceptable for the verifier, the verifier will abort the association. Otherwise, the verifier will continue with verifying the authentication token.

If the mechanism name or the algorithm name is missing in the authentication token, then the out-of-band agreed values for the mechanism resp. algorithm will be supported. If mechanism name or algorithm name is missing and no context is established prior to the commencement of the association establishment, then the association request shall be aborted or rejected.

Annex A (informative): Relationship to GSS-API

A.1 Introduction to GSS-API

The goal of GSS-API (*Generic Security Service Application Program Interface*) is stated in the abstract of STR-SEC-01.01 [17] as:

"The GSS-API defines security services and primitives at a level independent of underlying mechanism and programming language environment", and is to be completed by other, related specifications:

- documents defining specific parameter bindings for particular language environments;
- documents defining token formats, protocols, and procedures to be implemented in order to realize GSS-API services atop particular security mechanisms.

The latest version of GSS-API is GSS-API version 2 as defined in IETF RFC 2078 [18].

The use of GSS-API provides a **token based security solution** in the sense that all security related information that are exchanged in-line between two communicating entities are application transparent tokens (bit string). The information contained within the tokens, including the related token formats, will only be comprehensible internally within the modules that provide the GSS-API.

The term **GSS-API mechanism** has been defined to denote a particular set of underlying security mechanisms and related token formats to support these mechanisms. Currently, there are only two GSS-API mechanisms that have reached the maturity of being documented as IETF RFCs:

- SPKM (Simple Public Key Management) defined by IETF RFC 2025 [19];
- Kerberos v5 defined by IETF RFC 1964 [20].

In the future, further GSS-API mechanisms may appear.

A.2 Relationship between GSS-API and authentication

Whenever two peer entities wish to establish a secure connection through the use of GSS-API, the first step is for these entities to establish/negotiate a security context. This is performed through the exchange of a set of InitialContextTokens that contains all the security information that is required to establish a common security context.

Peer-to-peer entity authentication (unilateral or mutual) will be performed as part of this initial token exchange. Different authentication mechanisms are supported by the different GSS-API mechanisms and are as such an integral part of the particular GSS-API mechanism specifications.

A.3 GSS-API mechanisms supported by the present document

In order for a GSS-API mechanism to be conformant with the present document, the authentication mechanism(s) supported by that GSS-API mechanism should conform to one/several of the authentication mechanisms defined in clause 6.

Both the SPKM and the Kerberos v5 GSS-API mechanism are conformant with the present document. The SPKM supported authentication scheme is conformant with mechanism 7 and 8 defined in subclauses 6.3.1 and 6.3.2 while the Kerberos v5 supported authentication mechanism is conformant with mechanism 5 and 6 defined in subclauses 6.2.3 and 6.2.4.

A.4 Communication protocol mapping

It is outside the scope of the GSS-API related standards to define how the tokens are transferred between the calling and called party. However, for the purpose of peer-to-peer entity authentication for the use within TMN, the present document has selected ACSE as a protocol to use for this. This means that two no more than token exchanges can be used complete authentication/context establishment.

The GSS-API standard itself, STR-SEC-01.01 [17], does not place any restrictions on the number of token exchanges that can be used for authentication and context establishment. However, the use of ACSE for TMN forces the restriction that a GSS-API mechanism, in order to be applicable for TMN, must be able to support a one-way authentication scheme (unilateral authentication) or a two-way authentication scheme (unilateral or mutual authentication).

The protocol transfer syntax for inserting GSS-API InitialContextTokens into ACSE at the time of association establishment time is defined in subclause 7.2.2.

Annex B (informative): Algorithms

B.1 Hash functions

The following hash functions may be used for mechanisms 2 and 4:

Hash function	Reference
MD5	[13]
SHA-1	[14]
HMAC-MD5	[13], [24]
HMAC-SHA-1	[13], [24]
RIPEMD-160	[11]

B.2 Symmetric encipherment algorithms

The following symmetric encipherment algorithms may be used for the authentication mechanisms 5 and 6:

Symmetric encipherment algorithm	Reference
DES (in CBC mode)	[15], [10]
Triple DES (in CBC mode)	[15], [10]
IDEA	[21]
PNO	only for ETSI members

B.3 Public key algorithms

The following public key algorithms may be used for the authentication mechanisms 7 and 8:

Public key algorithm	Reference
RSA	[22]
DSA	[16]
Elliptic Curves/DSA like	[17]

Annex C (informative): Partial PICS for ACSE authentication information

The tables below give the protocol implementation conformance statements (PICS) for ACSE authentication information. Implementations which support the present document should support the requirements specified in the following tables.

Implementations which support unilateral authentication in the sending roles shall use the respective unilateral authentication tables.

Implementations which support mutual authentication in the sending or receiving roles shall use the respective mutual authentication tables.

This clause contains only partial PICS. An implementation shall combine these with the complete ACSE PICS.

C.1 Unilateral authentication parameter support

C.1.1 Sending (AARQ PDU)

Parameter name	Status	Syntax	Support	TVR
sender-acse-requirements	m	ACSE-requirements	Y	Functional unit for authentication
mechanism-name	m	Mechanism-name	Y	authentication-mechanism
calling-authentication-value	m	Authentication-Value	Y	authenticationToken

C.1.2 Receiving (AARQ PDU)

Parameter name	Status	Syntax	Support	TVR
sender-acse-requirements	m	ACSE-requirements	Y	Functional unit for authentication
mechanism-name	m	Mechanism-name	Y	authentication-mechanism
calling-authentication-value	m	Authentication-Value	Y	authenticationToken

C.2 Mutual authentication parameter support

C.2.1 Sending (AARQ PDU)

This table is the same as defined for unilateral authentication (see subclause C.1.1).

C.2.2 Receiving (AARQ PDU)

This table is the same as defined for unilateral authentication (see subclause C.1.2).

C.2.3 Sending (AARE PDU)

Parameter Name	Status	Syntax	Support	TVR
responder-acse-requirements	m	ACSE-requirements	Y	Functional unit for authentication
mechanism-name	m	Mechanism-name	Y	authentication-mechanism
responding-authentication-value	m	Authentication-Value	Y	authenticationToken

C.2.4 Receiving (AARE PDU)

Parameter Name	Status	Syntax	Support	TVR
responder-acse-requirements	m	ACSE-requirements	Y	Functional unit for authentication
mechanism-name	m	Mechanism-name	Y	authentication-mechanism
responding-authentication-value	m	Authentication-Value	Y	authenticationToken

Annex D (informative): Bibliography

- ITU-T Recommendation X.811 / ISO 10181-2: "Information Technology - Open Systems Interconnection - Security Frameworks for Open Systems: Authentication Framework", 4/95.
- ECMA-219: "Authentication and Privilege Attribute Security Application with related key distribution functions:

  Part 1: Overview and Functional Model

  Part 2: Security Information Objects

  Part 3: Service Definitions".
- RACE 2058 SAMSON: "TMN Security Profile", 8/95.
- RACE 2058 SAMSON: "Integration of Specific Security Architectures and Services", 8/95.
- Project P408 PAN-European TMN - Security Aspects, The long term security solution, 11/96.

History

Document history				
V1.1.1	April 1998	Public Enquiry	PE 9833:	1998-04-17 to 1998-08-14