BS ISO/IEC 9798-3:2019

BSI Standards Publication

IT Security techniques — Entity authentication

Part 3: Mechanisms using digital signature techniques

bsi.

National foreword

This British Standard is the UK implementation of ISO/IEC 9798-3:2019. It supersedes BS ISO/IEC 9798-3:1998+A1:2010, which is withdrawn.

The UK participation in its preparation was entrusted to Technical Committee IST/33/2, Cryptography and Security Mechanisms.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

Published by BSI Standards Limited 2019

ISBN 978 0 580 94987 6

ICS 35.030

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 31 January 2019.

Amendments/corrigenda issued since publication

Date	Text affected

BS ISO/IEC 9798-3:2019

INTERNATIONAL STANDARD

ISO/IEC 9798-3

Third edition
2019-01

IT Security techniques — Entity authentication —

Part 3: Mechanisms using digital signature techniques

Techniques de sécurité IT — Authentification d'entité —

Partie 3: Mécanismes utilisant des techniques de signature numériques

Reference number
ISO/IEC 9798-3:2019(E)

COPYRIGHT PROTECTED DOCUMENT

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html.

This document was prepared by Joint Technical Committee JTC 1, *Information Technology*, Subcommittee SC 27, *IT Security techniques*.

This third edition cancels and replaces the second edition (ISO/IEC 9798–3:1998), which has been technically revised. It also incorporates the amendment ISO/IEC 9798–3:1998/Amd 1:2010, and corrigenda ISO/IEC 9798–3:1998/Cor 1:2009 and ISO/IEC 9798–3:1998/Cor 2:2012. The main changes compared to the previous edition are as follows:

— all mechanisms have been technically revised to resolve security issues and make the mechanism secure by default;

— all mechanisms have been renamed and editorially improved to represent them more clearly;

— three additional mechanisms have been included using an on-line trusted third party;

— guidance to explain the security properties of the mechanisms and guide users in selecting the appropriate mechanism for their use case has been added (Annex B).

A list of all parts in the ISO/IEC 9798 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

INTERNATIONAL STANDARD **ISO/IEC 9798-3:2019(E)**

IT Security techniques — Entity authentication —

Part 3: Mechanisms using digital signature techniques

1 Scope

This document specifies entity authentication mechanisms using digital signatures based on asymmetric techniques. A digital signature is used to verify the identity of an entity.

Ten mechanisms are specified in this document. The first five mechanisms do not involve an on-line trusted third party and the last five make use of on-line trusted third parties. In both of these two categories, two mechanisms achieve unilateral authentication and the remaining three achieve mutual authentication.

Annex A defines the object identifiers assigned to the entity authentication mechanisms specified in this document.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 9798-1, *Information technology — Security techniques — Entity authentication — Part 1: General*

ISO/IEC 14888 (all parts), *Information technology — Security techniques — Digital signatures with appendix*

ISO/IEC 9796 (all parts), *Information technology — Security techniques — Digital signature schemes giving message recovery.*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
atomic transaction
transaction which cannot be split into multiple smaller transactions

3.2
claimant
entity which is or represents a principal for the purposes of authentication

[SOURCE: ISO/IEC 9798–1:2010, 3.6, modified — The Note to entry has been removed.]

3.3
digital signature
signature
data appended to, or a cryptographic transformation of, a data unit that allows the recipient of the data unit to verify the source and integrity of the data unit

3.4
entity authentication
corroboration that an entity is the one claimed

[SOURCE: ISO/IEC 9798–1:2010, 3.14]

3.5
mutual authentication
entity authentication (3.4) which provides both entities with assurance of each other's identity

[SOURCE: ISO/IEC 9798–1:2010, 3.18]

3.6
token
message consisting of data fields that are the output of a cryptographic function

3.7
trusted third party
security authority or its agent, trusted by other entities with respect to security related activities

[SOURCE: ISO/IEC 9798–1:2010, 3.38, modified — The Note to entry has been removed.]

3.8
unilateral authentication
entity authentication which provides one entity with assurance of the other's identity but not vice versa

[SOURCE: ISO/IEC 9798–1:2010, 3.39]

3.9
verifier
entity that requires to verify the identity of another entity

4 Symbols and abbreviated terms

The symbols and abbreviated terms given in ISO/IEC 9798–1 and the following shall apply.

$Cert_X$	certificate for entity X
I_X	representation of the identity of entity X, which is either i_X or $Cert_X$
i_X	string identifying entity X
M	data string that is input to a digital signature algorithm
P_X	public verification key associated with X
Res_X	result of verifying entity X's public key or public key certificate
SID^i_m	constant uniquely identifying the mechanism m and the signed string (number i) within the mechanism
$sS_X(M)$	signature on data string M with the private signing key of entity X. The signature shall be such that M can be recovered

$\frac{T_X}{N_X}$ time variant parameter used by entity X, either a sequence number N_X or a time stamp T_X

$X \| Y$ result of the concatenation of data items X and Y in the order specified. In cases where the result of concatenating two or more data items is signed as part of one of the mechanisms specified in this document, this result should be composed so that it can be uniquely resolved into its constituent data strings, i.e. so that there is no possibility of ambiguity in interpretation

NOTE Unique parsing of concatenated strings can be achieved in a variety of different ways, depending on the application. For example, it can be guaranteed by a) fixing the length of each of the substrings throughout the domain of use of the mechanism, or b) encoding the sequence of concatenated strings using a method that guarantees unique decoding, e.g., using the distinguished encoding rules defined in ISO/IEC 8825–1[3].

5 General

5.1 Time variant parameters

The mechanisms specified in this document use digital signatures to achieve unilateral or mutual entity authentication. Annex B provides guidance to explain the security properties of the mechanisms and guide users in selecting the appropriate mechanism for their use case.

To prevent valid authentication information from being accepted at a later time, time variant parameters such as time stamps, sequence numbers, or random numbers are used (see ISO/IEC 9798-1:2010, Annex B and the Note below).

If a time stamp or a sequence number is used, one pass is needed for unilateral authentication, while two passes are needed to achieve mutual authentication. If a challenge and response method employing random numbers is used, two passes are needed for unilateral authentication, while three or four passes (depending on the mechanism employed) are required to achieve mutual authentication.

NOTE The signing by one entity of a data block which has been manipulated by a second entity can be prevented by the first entity including its own random number in the data block which it signs. In this case, it is the unpredictability which prevents the signing of pre-defined data.

5.2 Tokens

Throughout this document, tokens are defined as:

$$\text{Token} = X_1 \| \cdots \| X_i \| sS_A(Y_1 \| \cdots \| Y_j).$$

In this document, the term "signed data" refers to the data string "$Y_1 \| \cdots \| Y_j$" used as input to the signature scheme and the term "unsigned data" refers to the data string "$X_1 \| \cdots \| X_i$".

Information contained in the unsigned data is, in general, not authenticated by the mechanisms in this document.

If information contained in the signed data of the token can be recovered from the signature [as is the case for signature schemes with message recovery, as specified in ISO/IEC 9796 (all parts)] or is already known to the verifier, then it does not need to be contained in the unsigned data of the token sent by the claimant.

When a signature scheme without message recovery is used, the signed data, M, should be inserted in the unsigned data right before the corresponding signature, i.e. $sS_X(M)$ is replaced by $M \| sS_X(M)$.

Parts of the signed data *M* that are already available to the recipient can be excluded from the unsigned version of *M*.

5.3 Use of text fields

All text fields specified in the following mechanisms are available for use in applications outside the scope of this document (they may be empty). Their relationship and contents depend on the specific application. See Annex C for information on the use of text fields.

6 Requirements

In the authentication mechanisms specified in this document, an entity to be authenticated corroborates its identity by demonstrating its knowledge of its private signature key. This is achieved by the entity using its private signature key to sign specific data. The signature can be verified by anyone using the entity's public verification key.

The authentication mechanisms have the following requirements:

a) A verifier shall possess the valid public key of the claimant, i.e. of the entity that the claimant claims to be.

   One way of obtaining a valid public key is by means of a certificate (see ISO/IEC 9798-1:2010, Annex C). The generation, distribution, and revocation of certificates are outside the scope of this document. Depending on the mechanism, a trusted third party may be used to distribute an authentic copy of the public key and its certificate. Another way of obtaining a valid public key is by a trusted courier.

   As the distribution of certificates is outside the scope of this document, the sending of certificates is optional in all mechanisms.

b) A claimant shall have a private signature key known and used only by the claimant.

c) The private signature key used in an implementation of one of the mechanisms specified in this document shall be distinct from keys used for any other purposes.

d) The data strings signed at various points in an authentication mechanism shall be composed so that they cannot be interchanged.

To help achieve requirement d), the mechanisms in this document include constants SID^i_m in the signed data.

NOTE The form of the constants, SID^i_m, is not specified in this document. However, in order to meet requirement d), they can be defined to include the following data elements:

— The object identifier as specified in Annex A, in particular identifying the ISO/IEC standard, the part number, and the authentication mechanism;

— A constant that uniquely identifies the signed string within the mechanism. This constant can be omitted in mechanisms that include only one signed string.

The recipient of a signature shall verify that the constant SID^i_m in the signed data is as expected.

If any of the above requirements is not satisfied, then the authentication process can be compromised or fail to complete successfully.

Annex A defines the object identifiers which shall be used to identify the entity authentication mechanisms specified in this document.

7 Mechanisms without an on-line trusted third party

7.1 Unilateral authentication

7.1.1 General

Unilateral authentication means that only one of the two entities is authenticated by use of the mechanism.

7.1.2 Mechanism UNI.TS — One-pass authentication

In this authentication mechanism, the claimant *A* initiates the process and is authenticated by the verifier *B*. Uniqueness and timeliness is controlled by generating and checking a time stamp or a sequence number (see ISO/IEC 9798-1:2010, Annex B).

The authentication mechanism is illustrated in Figure 1.

Figure 1 — One-pass unilateral authentication

The form of the token (Token*AB*), sent by the claimant *A* to the verifier *B* is:

$$\text{Token}AB = \text{Text}2 \| sS_A\left(SID^1_{\text{UNI.TS}} \| \frac{T_A}{N_A} \| i_B \| \text{Text}1\right),$$

where the claimant, *A*, uses either a sequence number, N_A, or a time stamp, T_A, as the time variant parameter. The choice depends on the technical capabilities of the claimant and the verifier as well as on the environment.

NOTE 1 The inclusion of the identifier i_B in the signed data of Token*AB* is necessary to prevent the token from being accepted by anyone other than the intended verifier.

NOTE 2 One application of this mechanism can be public key or certificate distribution (see ISO/IEC 9798-1:2010, Annex A).

a) *A* sends Token*AB* and, optionally, its identity, I_A, to *B*.

b) On receipt of the message containing Token*AB*, *B* performs the following steps:

   1) It checks the received identity, I_A, and determines whether this is trusted by verifying the certificate of *A*, matching it with a stored list of trusted entities or by some other means.

      NOTE 3 It can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

   2) It ensures that it is in possession of a valid public key of *A*.

   3) It verifies Token*AB* by verifying the signature of *A* contained in the token, by checking the *SID*, by checking the time stamp or the sequence number, and by checking that the value of the identifier field, (i_B), in the signed data of Token*AB* is equal to entity *B*'s distinguishing identifier.

7.1.3 Mechanism UNI.CR — Two-pass authentication

In this authentication mechanism, the claimant, *A*, is authenticated by the verifier, *B*, who initiates the process. Uniqueness and timeliness is controlled by generating and checking a random number, R_B (see ISO/IEC 9798-1:2010, Annex B).

The authentication mechanism is illustrated in Figure 2.

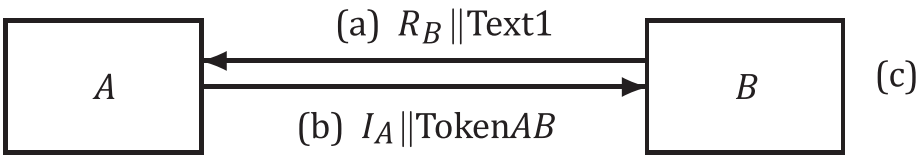

Figure 2 — Two-pass unilateral authentication

The form of the token (Token*AB*), sent by the claimant, *A*, to the verifier, *B*, is:

$$\text{Token}AB = \text{Text3} \parallel sS_A(SID1_{\text{UNI.CR}} \parallel R_A \parallel R_B \parallel i_B \parallel \text{Text2}).$$

NOTE 1 The inclusion of the identifier, i_B, in the signed data of Token*AB* prevents the token from being accepted by anyone other than the intended verifier (e.g., in a person-in-the-middle attack).

NOTE 2 The inclusion of the random number, R_A, in the signed part of Token*AB* prevents *B* from obtaining the signature of *A* on data chosen by *B* prior to the start of the authentication mechanism.

a) *B* sends a random number, R_B, and, optionally, a text field, Text1, to *A*.

b) *A* sends Token*AB* and, optionally, its identity, I_A, to *B*.

c) On receipt of the message containing Token*AB*, *B* performs the following steps:

   1) It checks the received identity, I_A, and determines whether this is trusted either by verifying the certificate of *A*, matching it with a stored list of trusted entities or by some other means.

      NOTE 3 It can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

   2) It ensures that it is in possession of a valid public key of *A*.

   3) It verifies Token*AB* by checking the signature of *A* contained in the token, by checking the *SID*, by checking that the random number, R_B, sent to *A* in step a), agrees with the random number contained in the signed data of Token*AB*, and by checking that the value of the identifier field, (i_B), in the signed data of Token*AB* is equal to *B*'s distinguishing identifier.

7.2 Mutual authentication

7.2.1 General

Mutual authentication means that the two communicating entities are authenticated to each other.

The two mechanisms described in 7.1.2 and 7.1.3 are extended in 7.2.2 and 7.2.3, respectively, to achieve mutual authentication. This is achieved by transmitting one further message resulting in two additional steps.

The mechanism specified in 7.2.4 uses four messages which do not need to be all sent consecutively. In this way, the authentication process can be speeded up.

7.2.2 Mechanism MUT.TS — Two-pass authentication

In this authentication mechanism, uniqueness and timeliness is controlled by generating and checking time stamps or sequence numbers (see ISO/IEC 9798-1:2010, Annex B).

The authentication mechanism is illustrated in Figure 3.

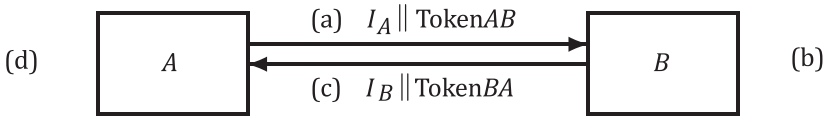

Figure 3 — Two-pass mutual authentication

The form of the token (Token*AB*), sent by *A* to *B*, is analogous to that specified in 7.1.2:

$$\text{Token}AB = \text{Text}2 \| sS_A \left(SID^1_{\text{MUT.TS}} \| \frac{T_A}{N_A} \| i_B \| \text{Text}1 \right).$$

The form of the token (Token*BA*), sent by *B* to *A*, is:

$$\text{Token}BA = \text{Text}4 \| sS_B \left(SID^2_{\text{MUT.TS}} \| \frac{T_B}{N_B} \| \frac{T_A}{N_A} \| i_A \| \text{Text}3 \right).$$

The choice of using either time stamps or sequence numbers in this mechanism depends on the technical capabilities of the claimant and the verifier as well as on the environment.

NOTE 1 The inclusion of identifiers, i_A and i_B, in the signed data of Token*BA* and Token*AB*, respectively, is necessary to prevent the tokens from being accepted by anyone other than the intended verifier.

NOTE 2 If $\frac{T_A}{N_A}$ were to be omitted in Token*BA*, the two messages of this mechanism are not bound together in any way, other than implicitly by timeliness; the mechanism involves independent use of mechanism 7.1.2 twice. The mechanism no longer achieves mutual authentication.

a) *A* sends Token*AB* and, optionally, its identity, I_A, to *B*.

b) On receipt of the message containing Token*AB*, *B* performs the following steps:

   1) It checks the received identity, I_A, and determines whether this is trusted either by verifying the certificate of *A*, matching it with a stored list of trusted entities or by some other means.

      NOTE 3 It can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

   2) It ensures that it is in possession of a valid public key of *A*.

   3) It verifies Token*AB* by verifying the signature of *A* contained in the token, by checking the *SID*, by checking the time stamp or the sequence number, and by checking that the value of the identifier field, (i_B), in the signed data of Token*AB* is equal to entity *B*'s distinguishing identifier.

c) *B* sends Token*BA* and, optionally, its identity, I_B, to *A*.

d) On receipt of the message containing Token*BA*, *A* performs the following steps:

   1) It checks the received identity, I_B, and determines whether this is trusted either by verifying the certificate of *B*, matching it with a stored list of trusted entities or by some other means.

      NOTE 4 It can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

2) It ensures that the received identity, I_B, corresponds to i_B included in TokenAB.

3) It ensures that it is in possession of a valid public key of B.

4) It verifies TokenBA by verifying the signature of B contained in the token, by checking the *SID*, by checking the time stamp or the sequence number, and by checking that the value of the identifier field, (i_A), in the signed data of TokenBA is equal to entity A's distinguishing identifier.

5) A verifies that the $\frac{T_A}{N_A}$ received in TokenBA is identical to the one sent in TokenAB in step a).

7.2.3 Mechanism MUT.CR — Three-pass authentication

In this authentication mechanism, uniqueness and timeliness is controlled by generating and checking random numbers (see ISO/IEC 9798-1:2010, Annex B).

The authentication mechanism is illustrated in Figure 4.

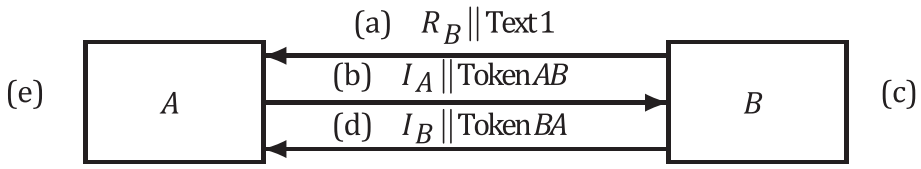

Figure 4 — Three-pass mutual authentication

The tokens are of the following form:

$$\text{Token}AB = \text{Text3} \| sS_A\left(SID^1_{\text{MUT.CR}} \| R_A \| R_B \| i_B \| \text{Text2}\right);$$

$$\text{Token}BA = \text{Text5} \| sS_B\left(SID^2_{\text{MUT.CR}} \| R'_B \| R_A \| i_A \| \text{Text4}\right).$$

NOTE 1 When i_B or i_A are omitted from TokenAB or TokenBA, respectively, A cannot conclude B is intending to authenticate to A (and vice versa). Additionally, agreement on Text2 and Text4 cannot be guaranteed.

NOTE 2 The inclusion of the random number, R_A, in the signed part of TokenAB prevents B from obtaining the signature of A on data chosen by B prior to the start of the authentication mechanism. The same holds for the random number, R'_B, in the signed part of TokenBA. R'_B can be set to R_B, but, in this case, A can obtain a signature from B on data chosen prior to sending TokenAB.

a) B sends a random number, R_B, and, optionally, a text field, Text1, to A.

b) A sends TokenAB and, optionally, its identity, I_A, to B.

c) On receipt of the message containing TokenAB, B performs the following steps:

1) It checks the received identity, I_A, and determines whether this is trusted either by verifying the certificate of A, matching it with a stored list of trusted entities or by some other means.

NOTE 3 It can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

2) It ensures that it is in possession of a valid public key of A.

3) It verifies TokenAB by checking the signature of A contained in the token, by checking the *SID*, by checking that the random number, R_B, sent to A in step a), agrees with the random number contained in the signed data of TokenAB, and by checking that the value of the identifier field, (i_B), in the signed data of TokenAB is equal to B's distinguishing identifier.

d) B sends TokenBA and, optionally, its identity, I_B, to A.

e) On receipt of the message containing Token*BA*, *A* performs the following steps:

1) It checks the received identity, I_B, and determines whether this is trusted either by verifying the certificate of *B*, matching it with a stored list of trusted entities or by some other means.

NOTE 4 It can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

2) It ensures that the received identity, I_B, corresponds to i_B included in Token*AB*.

3) It ensures that it is in possession of a valid public key of *B*.

4) It verifies Token*BA* by checking the signature of *B* contained in the token, by checking the *SID*, by checking that the random number, R_A, sent to *B* in step b), agrees with the random number contained in the signed data of Token*BA*, and by checking that the value of the identifier field, (i_A), in the signed data of Token*BA* is equal to *A*'s distinguishing identifier.

7.2.4 Mechanism MUT.CR.par — Two-pass parallel authentication

In this mechanism, authentication is carried out in parallel. Uniqueness and timeliness is controlled by generating and checking random numbers (see ISO/IEC 9798-1:2010, Annex B).

The authentication mechanism is illustrated in Figure 5.

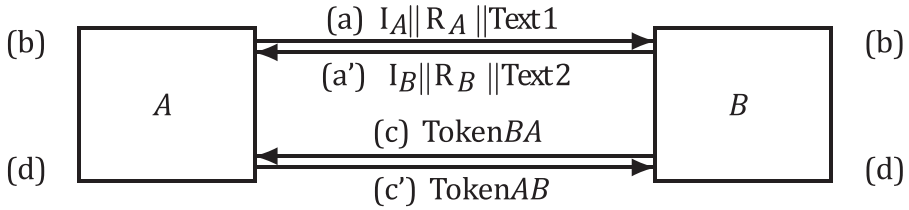

Figure 5 — Two-pass parallel authentication

The tokens are similar to those of 7.1.3:

$$\text{Token}AB = \text{Text}4 \| sS_A\left(SID^1_{\text{MUT.CR.par}} \| R_A \| R_B \| i_B \| \text{Text}3\right),$$

$$\text{Token}BA = \text{Text}6 \| sS_B\left(SID^1_{\text{MUT.CR.par}} \| R_B \| R_A \| i_A \| \text{Text}5\right).$$

NOTE 1 The random number, R_A, is present in Token*AB* to prevent *B* from obtaining the signature of *A* on data chosen by *B* prior to the start of the authentication mechanism. For similar reasons, the random number, R_B, is present in Token*BA*. Depending on the relative time of receipt of the messages sent in steps (1) and (1)', one of the parties can know the random number of the other party when choosing its random number. If this is undesirable, both parties can insert an additional random number, R'_A and R'_B, in the text fields Text3 of Token*AB* and Text5 of Token*BA*, respectively.

NOTE 2 Both signatures in the mechanism have the same identifier, $SID^1_{\text{MUT.CR.par}}$, since the order of messages is not fixed.

a) *A* sends a random number, R_A, and, optionally, its identity, I_A, and optionally a text field, Text1, to *B*.

a') *B* sends a random number, R_B, and, optionally, its identity, I_B, and optionally a text field, Text2, to *A*.

b) *A* and *B* each perform the following steps:

1) Each of them checks the received identity, I_X, and determines whether this is trusted either by verifying the certificate of the other entity, matching it with a stored list of trusted entities or by some other means.

NOTE 3 Each of them can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

2) Each of them ensures that it is in possession of a valid public key of the other entity.

c) *A* sends Token*AB* to *B*, where Token*AB* contains, i_B, corresponding to the identity of the entity that is considered trusted by *A* in step b).

c') *B* sends Token*BA* to *A*, where Token*BA* contains, i_A, corresponding to the identity of the entity that is considered trusted by *B* in step b).

d) *A* and *B* each perform the following steps:

1) Each of them verifies the received token by checking the signature contained in the token [by using the public key from step b)] and by checking the *SID*.

2) Each of them checks that the random number, which it previously sent to the other entity, agrees with the first random number contained in the signed data of the token received.

3) Each of them checks that the random number it previously received from the other entity [n step a)] agrees with second the random number contained in the signed data of the token received.

4) Each of them checks that the value of the identity field, (i_X), contained in the signed data of the received token corresponds to its own identity.

8 Mechanisms involving an on-line trusted third party

8.1 General

Implementations of the mechanisms in this clause shall use one of the signature schemes specified in ISO/IEC 14888 (all parts) or ISO/IEC 9796 (all parts).

In the specification of the mechanisms in this clause, the form of tokens and text fields follows the description given in Clauses 4 and 5. In addition, the values of the fields Res_A, Res_B, Status and Failure in the mechanisms specified in this clause shall have the following forms:

— Res_A = ($Cert_A$ || Status), (i_A ||P_A) or Failure;

— Res_B = ($Cert_B$ || Status), (i_B ||P_B) or Failure;

— *Status* = True or False. The value of the field shall be set to False if the certificate validation (e.g. according to ISO/IEC 9594-8,[4] ITU-T X.509[7] or the security policy of the domain in which the TP is residing) fails. Otherwise it shall be set to True.

— Failure:Res_X (where $X \in \{A,B\}$) will be set to Failure if neither a public key nor a certificate of entity *X* can be found by *TP*.

In the mechanisms of this clause, if TP knows the mapping between identity *X* and P_X (where $X \in \{A,B\}$), then it shall set $I_X = i_X$; otherwise, it shall set $I_X = Cert_X$, and *X* shall be set equal to the collection of distinguished identity fields in $Cert_X$. If either *X* or $Cert_X$ is permitted to be used as an identity, then there should be a pre-arranged means to allow TP to distinguish the two types of identity indications. The value of Res_X (where $X = \{A,B\}$) shall be determined according to Table 1.

Table 1 — Value of Res_X

Field	Choice 1	Choice 2
I_X	i_X	$Cert_X$
Res_X	(i_X || P_X) or Failure	($Cert_X$ || Status) or Failure

8.2 Unilateral authentication

8.2.1 General

The authentication mechanisms in this subclause require the two entities *A* (or *B*) to validate the other's public key using an on-line trusted third party (with distinguishing identifier *TP*). This trusted third party shall have the capability to verify the authenticity of the public key of *A* (or *B*). The entities *A* (or *B*) shall possess a reliable copy of the public key of *TP*.

This subclause specifies two four-pass authentication mechanisms, both of which achieve unilateral authentication between entities *A* and *B*. Furthermore, the mechanisms in this subclause provide entity authentication of the *TP* as well as origin authentication and non-replay of the verification results. The four-pass authentication is an atomic transaction.

Implementations of the mechanisms shall use one of the signature schemes specified in ISO/IEC 14888 (all parts) or ISO/IEC 9796 (all parts).

8.2.2 Mechanism TP.UNI.1 — Four-pass authentication (initiated by *A*)

In this authentication mechanism, the claimant *B* is authenticated by the verifier *A* who initiates the process, using an on-line trusted third party (with distinguishing identifier *TP*). *TP* shall have the capability to verify the authenticity of the public key of *B*. The entity *A* shall possess a reliable copy of the public key of *TP*.

This authentication mechanism is illustrated in Figure 6.

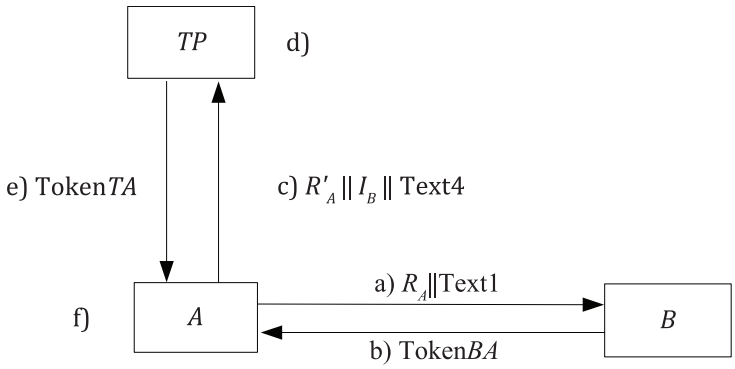

Figure 6 — Four-pass authentication (initiated by *A*)

The tokens shall be created as follows:

$$\text{Token}BA = \text{Text2} \| sS_B\left(SID^1_{\text{TP.UNI.1}} \| R_B \| R_A \| i_A \| \text{Text3}\right);$$

$$\text{Token}TA = \text{Text5} \| sS_T\left(SID^1_{\text{TP.UNI.1}} \| R'_A \| Res_B \| \text{Text6}\right).$$

The mechanism is performed as follows:

a) *A* sends a random number, R_A, and, optionally, a text field, Text1, to *B*.

b) *B* sends the token Token*BA* to *A*.

c) *A* sends a random number, R'_A, I_B. and, optionally, a text field, Text4, to *TP*.

d) On receipt of the message in step c) from *A*, *TP* performs the following steps. If $I_B = i_B$, *TP* retrieves P_B. If $I_B = Cert_B$, *TP* checks the validity of $Cert_B$. The process of certificate verification by *TP* can

require protection from denial-of-service attacks. The specification of mechanisms to be used to provide such protection is outside of the scope of this document.

e) *TP* sends Token*TA* to *A*. The fields Res_B in Token*TA* shall be: the certificate of *B* and its status, the distinguishing identifier of *B* and its public key, or an indication of Failure.

f) On receipt of the message in step e) from *TP*, *A* performs the following steps:

1) Verify Token*TA* by checking the signature of *TP* contained in the token, by checking the *SID*, and by checking that the random number R'_A, sent to *TP* in Step c), is the same as the random number, R'_A, contained in the signed data of Token*TA*, and by checking Res_B is not Failure.

NOTE It can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

2) Retrieve the public key of *B* from the message, verify Token*BA* received in step b) by checking the signature of *B* contained in the token, by checking the *SID* and checking that the value of identifier field, (i_A), in the signed data of Token*BA* is equal to *A*'s distinguishing identifier, and then check that the random number, R_A, sent to *B* in step a), is the same as the random number, R_A, contained in Token*BA*.

8.2.3 Mechanism TP.UNI.2 — Four-pass authentication (initiated by *B*)

In this authentication mechanism, the claimant *A* is authenticated by the verifier *B* who initiates the process, using an on-line trusted third party (with distinguishing identifier *TP*). *TP* shall have the capability to verify the authenticity of the public key of *A*. The entity *B* shall possess a reliable copy of the public key of *TP*.

This authentication mechanism is illustrated in Figure 7.

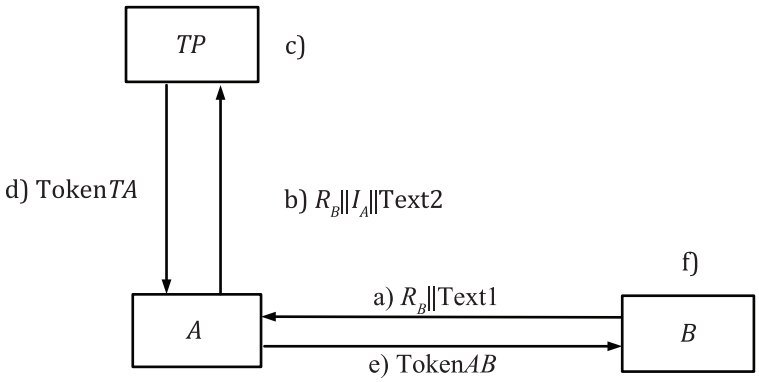

Figure 7 — Four-pass authentication (initiated by *B*)

The tokens shall be created as follows:

$$\text{Token}TA = \text{Text3} \,||\, sS_T\left(SID^1_{\text{TP.UNI.2}} \,||\, R_B \,||\, Res_A \,||\, \text{Text4}\right);$$

$$\text{Token}AB = \text{Text5} \,||\, \text{Token}TA \,||\, sS_A\left(SID^2_{\text{TP.UNI.2}} \,||\, R_A \,||\, R_B \,||\, i_B \,||\, \text{Text6}\right).$$

The mechanism is performed as follows:

a) *B* sends a random number, R_B, and, optionally, a text field, Text1, to *A*.

b) *A* sends, R_B, I_A, and, optionally, a text field, Text2, to *TP*.

c) On receipt of the message in Step b) from *A*, *TP* performs the following steps: If $\text{I}_\text{A} = i_A$, *TP* retrieves P_A. If I_A = $Cert_A$, TP checks the validity of $Cert_A$. The process of certificate verification by *TP* can

require protection from denial-of-service attacks. The specification of mechanisms to be used to provide such protection is outside of the scope of this document.

d) *TP* sends Token*TA* to *A*. The fields Res_A in Token*TA* shall be: the certificate of *A* and its status, the distinguishing identifier of *A* and its public key, or an indication of Failure.

e) *A* sends the token Token*AB* to *B*.

f) On receipt of the message in step e) from *A*, *B* performs the following steps:

1) Verify the signature of *TP* in Token*TA* by checking the signature of *TP* contained in the token, by checking the *SID* and by checking that the random number, R_B, sent to *A* in step a), is the same as the random number, R_B, contained in the signed data of TP_B of Token*TA*, and by checking Res*A* is not Failure.

NOTE It can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

2) Retrieve the public key of *A* from the message, verify Token*AB* by checking the signature of *A* contained in the token, by checking *SID*, and checking that the value of identifier field, (i_B), in the signed data of Token*AB* is equal to *B*'s distinguishing identifier, and then check that the random number, R_B, sent to *A* in step a), is the same as the random number, R_B, contained in the signed data of *A* of Token*AB*.

8.3 Mutual authentication

8.3.1 General

The authentication mechanisms in this subclause require the two entities *A* and *B* to validate each other's public keys using one or two on-line trusted third parties.

If only a single on-line trusted third party is used, it has a distinguishing identifier denoted by *TP.*

If two on-line trusted third parties are used, their distinguishing identifiers are denoted by TP_A and TP_B respectively. The authenticity of the public key of *A* is verified only by TP_A, and the authenticity of the public key of *B* is verified only by TP_B. Entity *A* trusts TP_A (*A* accepts any assertion signed by TP_A as valid) and shall possess a reliable copy of the public key of corresponding TP_A. Entity *B* trusts TP_B (*B* accepts any assertion signed by TP_B as valid) and shall possess a reliable copy of the public key of corresponding TP_B. TP_A and TP_B trust each other. TP_A has a reliable copy of TP_B's public key and TP_B has a reliable copy of TP_A's public key.

This subclause specifies two five-pass and one seven-pass authentication mechanisms, all of which achieve mutual authentication between entities *A* and *B*. Furthermore, these mechanisms provide entity authentication of the *TP*, TP_A, or TP_B as well as origin authentication and non-replay of the verification results. The five-pass and seven-pass authentication mechanisms are atomic transaction.

NOTE The mechanisms in this subclause are intended to be used in a closed environment, where all entities share the same *TP* and possess a reliable copy of its public key. If Option 1 of the mechanisms is used, *TP* only provides a certificate validation service. In case Option 2 of the mechanisms is used, besides the certificate validation service, *TP* can also provide an authorization service for the entities *A* and *B* to communicate to each other.

If Option 1 of the mechanisms is used in an environment in which *B* (or *A*) should know that *TP* is validating *A*'s (or *B*'s) credentials for *B* (or *A*), the Text in the signatures of *TP* in Option 1 should include I_A (or I_B) respectively. More specifically, the *TP* can make I_A as part of the text in the first signature of Token*TA* and I_B similarly in the second signature of Token*TA*. In this case, *A*, *B* and *TP* should be in consensus on the format and location of the value I_A (or I_B) included in such a text.

8.3.2 Mechanism TP.MUT.1 — Five-pass authentication (initiated by *A*)

In this authentication mechanism, uniqueness and timeliness is controlled by generating and checking a random number (see ISO/IEC 9798-1:2010, Annex B).

This authentication mechanism is illustrated in Figure 8.

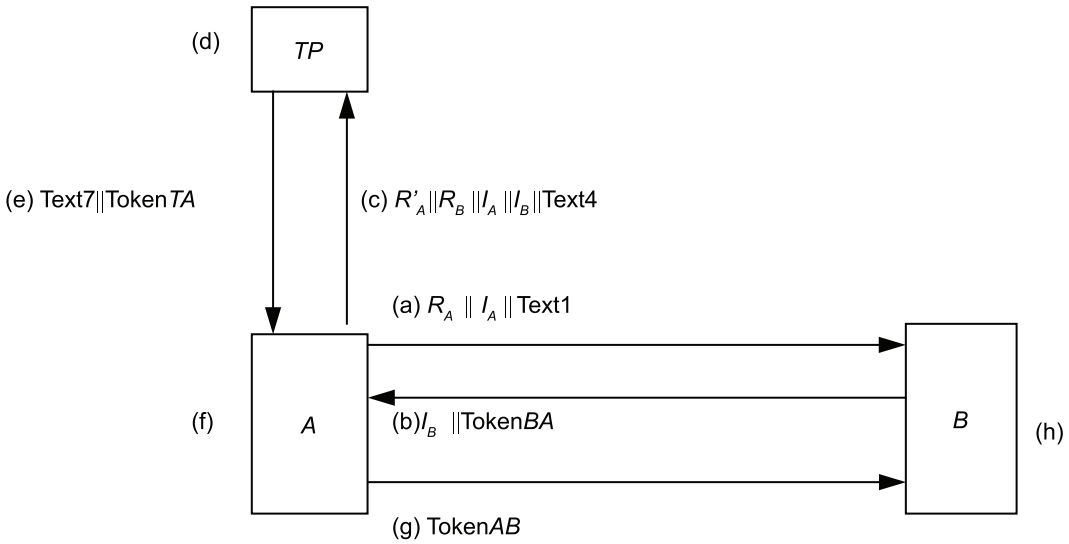

Figure 8 — Five-pass authentication (initiated by *A*)

The tokens shall be created in accordance with one of the following two options.

Option 1:

— $\text{Token}BA = \text{Text3} \| sS_B\left(SID^1_{\text{TP.MUT.1–1}} \| i_B \| R_A \| R_B \| i_A \| \text{Text2}\right)$;

— $\text{Token}TA = sS_T\left(SID^2_{\text{TP.MUT.1–1}} \| R'_A \| Res_B \| \text{Text6}\right) \| sS_T\left(SID^3_{\text{TP.MUT.1–1}} \| R_B \| Res_A \| \text{Text5}\right)$;

— $\text{Token}AB = \text{Text9} \| sS_T\left(SID^3_{\text{TP.MUT.1–1}} \| R_B \| Res_A \| \text{Text5}\right) \| sS_A\left(SID^4_{\text{TP.MUT.1–1}} \| R_B \| R''_A \| i_B \| i_A \| \text{Text8}\right)$.

Option 2:

— $\text{Token}BA = \text{Text3} \| sS_B\left(SID^1_{\text{TP.MUT.1–2}} \| i_B \| R_A \| R_B \| i_A \| \text{Text2}\right)$;

— $\text{Token}TA = sS_T\left(SID^2_{\text{TP.MUT.1–2}} \| R'_A \| R_B \| Res_A \| Res_B \| \text{Text5}\right)$;

— $\text{Token}AB = \text{Text9} \| \text{Token}TA \| sS_A\left(SID^3_{\text{TP.MUT.1–2}} \| R_B \| R''_A \| i_B \| i_A \| \text{Text8}\right)$.

NOTE 1 Implementations of this mechanism can support one or both of the above options.

NOTE 2 The inclusion of the random number, R_A, in the signed part of Token*AB* prevents *B* from obtaining the signature of *A* on data chosen by *B* prior to the start of the authentication mechanism. The same holds for the random number, R_B, in the signed part of Token*BA*.

The mechanism is performed as follows:

a) *A* sends a random number, R_A, its identity, I_A, and, optionally, a text field, Text1, to *B*.

b) *B* sends the token Token*BA* and I_B to *A*.

c) *A* sends a random number, R'_A, together with R_B, I_A, I_B and, optionally, a text field, Text4, to *TP*.

d) On receipt of the message in Step c) from *A*, *TP* performs the following steps. If $I_A = i_A$ and $I_B = i_B$, *TP* retrieves P_A and P_B. If $I_A = Cert_A$ and $I_B = Cert_B$, *TP* checks the validity of $Cert_A$ and $Cert_B$. The process of certificate verification by *TP* can require protection from denial-of-service attacks. The specification of mechanisms to be used to provide such protection is outside of the scope of this document.

e) Then *TP* sends Token*TA* and, optionally, a text field, Text7, to *A*. The fields Res_A and Res_B in Token*TA* shall be: the certificates of *A* and *B* and their status, the distinguishing identifiers of *A* and *B* and their public keys, or an indication of Failure.

f) On receipt of the message in step e) from *TP*, *A* performs the following steps:

1) Verify Token*TA* by checking the signature of *TP* contained in the token, by checking the *SID* and by checking that the random number, R'_A, sent to *TP* in step c), is the same as the random number, R'_A, contained in the signed data of Token*TA*.

The signature containing Res_B should be checked. The signature without Res_B may be checked optionally. If the signature without Res_B is checked, R_B should be checked.

NOTE 3 *A* can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

2) Retrieve the public key of *B* from the message, verify Token*BA* received in step b) by checking the signature of *B* contained in the token, by checking the *SID* and checking that the value of identifier field, (i_A), in the signed data of Token*BA* is equal to *A*'s distinguishing identifier, and then check that the random number, R_A, sent to *B* in step a), is the same as the random number, R_A, contained in Token*BA*.

g) *A* sends Token*AB* to *B*.

h) On receipt of the message in step g) from *A*, *B* performs the following steps:

1) Verify the signature of *TP* in either Token*TA* (entity *A*) or Token*AB* (entity *B*), check the *SID*, and check that the random number, R_B, sent to *A* in step b), is the same as the random number, R_B, contained in the signed data of Token*TA*.

NOTE 4 *B* can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

2) Retrieve the public key of *A* from the message, verify Token*AB* by checking the signature of *A* contained in the token, checking the *SID*, and checking that the value of identifier field, (i_B), in the signed data of Token*AB* is equal to *B*'s distinguishing identifier, and then check that the random number, R_B, contained in the signed data of Token*AB* is equal to the random number, R_B, sent to *A* in Step b).

8.3.3 Mechanism TP.MUT.2 — Five-pass authentication (initiated by *B*)

In this authentication mechanism, uniqueness and timeliness is controlled by generating and checking a random number (see ISO/IEC 9798-1:2010, Annex B). This authentication mechanism is illustrated in Figure 9.

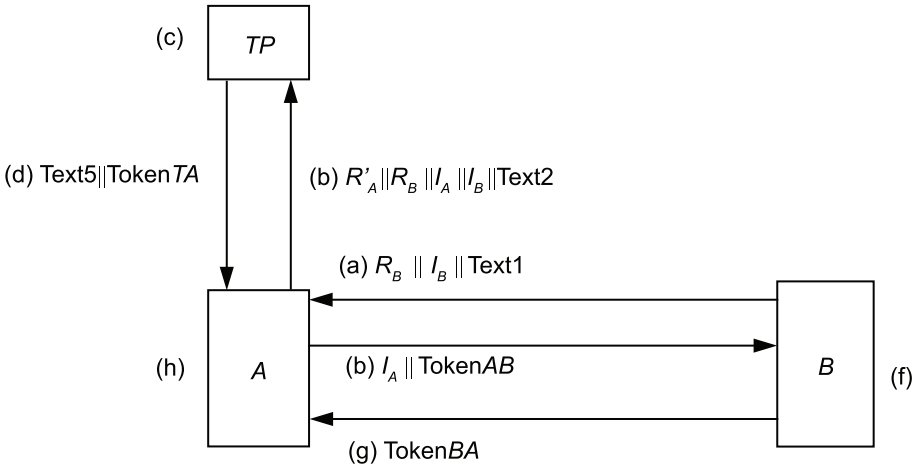

Figure 9 — Five-pass authentication (initiated by *B*)

The tokens shall be created in accordance with one of the following two options.

Option 1:

— $\text{Token}TA = sS_T\left(SID^1_{\text{TP.MUT.2–1}} \| R'_A \| Res_B \| \text{Text}4\right) \| sS_T\left(SID^2_{\text{TP.MUT.2–1}} \| R_B \| Res_A \| \text{Text}3\right)$;

— $\text{Token}AB = \text{Text}7 \| sS_T\left(SID^2_{\text{TP.MUT.2–1}} \| R_B \| Res_A \| \text{Text}3\right) \| sS_A\left(SID^3_{\text{TP.MUT.2–1}} \| R_B \| R_A \| i_B \| i_A \| \text{Text}6\right)$;

— $\text{Token}BA = \text{Text}9 \| sS_B\left(SID^4_{\text{TP.MUT.2–1}} \| i_A \| R_A \| R'_B \| i_B \| \text{Text}8\right)$.

Option 2:

— $\text{Token}TA = sS_T\left(SID^1_{\text{TP.MUT.2–2}} \| R'_A \| R_B \| Res_A \| Res_B \| \text{Text}3\right)$;

— $\text{Token}AB = \text{Text}7 \| \text{Token}TA \| sS_A\left(SID^2_{\text{TP.MUT.2–2}} \| R_B \| R_A \| i_B \| i_A \| \text{Text}6\right)$;

— $\text{Token}BA = \text{Text}9 \| sS_B\left(SID^3_{\text{TP.MUT.2–2}} \| R_A \| R'_B \| i_A \| i_B \| \text{Text}8\right)$.

NOTE 1 Implementations of this mechanism can support one or both of the above options.

The mechanism is performed as follows:

a) *B* sends a random number, R_B, its identity, I_B, and, optionally, a text field, Text1, to *A*.

b) *A* sends a random number, R'_A, together with R_B, I_A, I_B and, optionally, a text field, Text2, to *TP*.

   On receipt of the message in step b) from *A*, *TP* performs the following steps. If $I_A = i_A$ and $I_B = i_B$, *TP* retrieves P_A and P_B. If $I_A = Cert_A$ and $I_B = Cert_B$, *TP* checks the validity of $Cert_A$ and $Cert_B$. The process of certificate verification by *TP* can require protection from denial-of-service attacks. The specification of mechanisms to be used to provide such protection is outside of the scope of this document.

c) Then *TP* sends Token*TA* and, optionally, a text field, Text5, to *A*. The fields Res_A and Res_B in Token*TA* shall be: the certificates of *A* and *B* and their status, the distinguishing identifiers of *A* and *B* and their public keys or an indication of Failure.

d) A sends the token Token*AB* and I_A to *B*.

e) On receipt of the message in step e) from *A*, *B* performs the following steps:

1) Verify the signature of *TP* in Token*AB* by checking the signature of *TP* contained in the token, checking the *SID*, and by checking that the random number, R_B, sent to *A* in step a), is the same as the random number, R_B, contained in the signed data of *TP* of Token*AB*.

NOTE 2 It can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

2) Retrieve the public key of *A* from the message, verify Token*AB* by checking the signature of *A* contained in the token, checking the *SID*, and checking that the value of the identifier field, (i_B), in the signed data of Token*AB* is equal to *B*'s distinguishing identifier, and then check that the random number, R_B, sent to *A* in step a), is the same as the random number, R_B, contained in the signed data of *A* of Token*AB*.

f) *B* sends Token*BA* to *A*.

g) On receipt of the message in step g) from *B*, *A* performs the following steps:

1) Verify Token*TA* in the message from step d) by checking the signature of *TP* contained in the token, and by checking that the random number, R'_A, sent to *TP* in step b), is the same as the random number, R'_A, contained in the signed data of Token*TA*.

The signature containing Res_B should be checked. The signature without Res_B may be checked optionally.

NOTE 3 It can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

2) Retrieve the public key of *B* from the message of step d), verify Token*BA* by checking the signature of *B* contained in the token and checking that the value of the identifier field, (i_A), in the signed data of Token*BA* is equal to *A*'s distinguishing identifier, and then check that the random number, R_A, contained in the signed data of Token*BA* is equal to the random number, R_A, sent to *B* in step e).

8.3.4 Mechanism TP.MUT.3 — Seven-pass authentication (initiated by *B*)

This authentication mechanism has seven message passes, and makes use of two on-line trusted third parties (with distinguishing identifiers TP_A and TP_B).

In this authentication mechanism, uniqueness and timeliness is controlled by generating and checking a random number (see ISO/IEC 9798-1:2010, Annex B).

This authentication mechanism is illustrated in Figure 10.

The tokens shall be as follows:

— $\text{Token}TPAB = sS_{TPA}\left(SID^1_{\text{TP.MUT.3}} \| Res_A \| I_B \| R_B \| \text{Text3}\right)$;

— $\text{Token}TPBA = sS_{TPB}\left(SID^2_{\text{TP. MUT.3}} \| Res_A \| R_B \| \text{Text4}\right) \| sS_{TPB}\left(SID^3_{\text{TP.MUT.3}} \| Res_B \| R_{TPA} \| \text{Text5}\right)$;

— $\text{Token}TA = sS_{TPA}\left(SID^4_{\text{TP. MUT.3}} \| Res_B \| R'_A \| \text{Text6}\right) \| sS_{TPB}\left(SID^2_{\text{TP.MUT.3}} \| Res_A \| R_B \| \text{Text4}\right)$;

— $\text{Token}AB = sS_{TPB}\left(SID^2_{\text{TP. MUT.3}} \| Res_A \| R_B \| \text{Text4}\right) \| sS_A\left(SID^5_{\text{TP.MUT.3}} \| R_B \| R_A \| i_B \| i_A \| \text{Text7}\right)$;

— $\text{Token}BA = sS_B\left(SID^6_{\text{TP.MUT.3}} \| R_A \| R_B \| i_A \| i_B \| \text{Text8}\right)$.

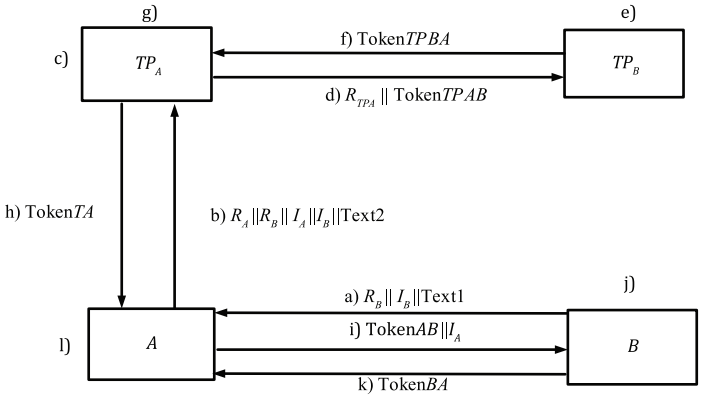

Figure 10 — Seven-pass authentication with two TTPs (initiated by *B*)

The mechanism is performed as follows:

a) *B* sends a random number, R_B, its identity, I_B, and, optionally, a text field, Text1, to *A*.

b) *A* sends a random number, R'_A, together with R_B, I_A, I_B and, optionally, a text field, Text2, to TP_A.

c) On receipt of the message in step b) from *A*, TP_A performs the following steps. If $I_A = i_A$, TP_A retrieves P_A. If $I_A = Cert_A$, TP_A checks the validity of $Cert_A$. The process of certificate verification by TP_A can require protection from denial-of-service attacks. The specification of mechanisms to be used to provide such protection is outside of the scope of this document.

d) TP_A sends R_{TPA} and Token*TPAB* to TP_B.

e) On receipt of the message in step d) from TP_A, TP_B performs the following steps:

   1) Verify the signature of TP_A in Token*TPAB* by checking the signature of TP_A contained in the token.

   2) If $I_B = i_B$, TP_B retrieves P_B. If $I_B = Cert_B$, TP_B checks the validity of $Cert_B$. The process of certificate verification by TP_B can require protection from denial-of-service attacks. The specification of mechanisms to be used to provide such protection is outside of the scope of this document.

f) TP_B sends Token*TPBA* to TP_A.

g) On receipt of the message in Step f) from TP_B, TP_A verify Token*TPBA* by checking the signature of TP_B contained in the token, and by checking that the random number, R_{TPA}, sent to TP_B in step d), is the same as the random number, R_{TPA}, contained in the signed data of Token*TPBA*.

h) TP_A sends Token*TA* to *A*. The fields Res_A and Res_B in Token*TA* shall be: the certificates of *A* and *B* and their status, the distinguishing identifiers of *A* and *B* and their public keys, or an indication of Failure.

i) *A* sends the token Token*AB* and I_A to *B*.

j) On receipt of the message in step i) from *A*, *B* performs the following steps:

   1) Verify the signature of TP_B in Token*AB* by checking the signature of TP_B contained in the token, and by checking that the random number, R_B, sent to *A* in step a), is the same as the random number, R_B, contained in the signed data of TP_B of Token*AB*.

      NOTE 1 It can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

   2) Retrieve the public key of *A* from the message, verify Token*AB* by checking the signature of *A* contained in the token and checking that the value of the identifier field (*B*) in the signed data of Token*AB* is equal to *B*'s distinguishing identifier, and then check that the random number,

R_B, sent to *A* in Step a), is the same as the random number, R_B, contained in the signed data of *A* of Token*AB*.

k) *B* sends Token*BA* to *A*.

l) On receipt of the message in step k) from *B*, *A* performs the following steps:

1) Verify Token*TA* by checking the signature of TP_A contained in the token, and by checking that the random number, R'_A, sent to TP_A in step b), is the same as the random number, R'_A, contained in the signed data of Token*TA*.

NOTE 2 It can also check whether the received identity is equal to its own identity. In many applications, authenticating an entity against itself is considered a security issue.

2) Retrieve the public key of *B* from the message of step h), verify Token*BA* by checking the signature of *B* contained in the token and checking that the value of identifier field, (*A*), in the signed data of Token*BA* is equal to *A*'s distinguishing identifier, and then check that the random number, R'_A, contained in the signed data of Token*BA* is equal to the random number, *I*, sent to *B* in step i).

Annex A
(normative)

Object Identifiers

A.1 Formal definition

```
EntityAuthenticationMechanisms-3 {
  iso(1) standard(0) e-auth-mechanisms(9798) part3(3)
    asn1-module(0) object-identifiers(0) }
DEFINITIONS EXPLICIT TAGS ::= BEGIN
-- EXPORTS All; --
-- IMPORTS None; --
OID ::= OBJECT IDENTIFIER -- alias
-- Synonyms --
is9798-3 OID ::= { iso(1) standard(0) e-auth-mechanisms(9798) part3(3)
 }
mechanism OID ::= { is9798-3 mechanisms-2019(2) }
-- mechanisms not involving a trusted third party --
nottp-mechanism OID ::= { mechanism nottp(1) }
nottp-uni-mechanism OID ::= { nottp-mechanism uni(1) }
nottp-mut-mechanism OID ::= { nottp-mechanism mut(2) }
uni-ts OID ::= { nottp-uni-mechanism 1 }
uni-cr OID ::= { nottp-uni-mechanism 2 }
mut-ts OID ::= { nottp-mut-mechanism 1 }
mut-cr OID ::= { nottp-mut-mechanism 2 }
mut-cr-Parallel OID ::= { nottp-mut-mechanism 3 }
-- mechanisms involving a trusted third party –
ttp-mechanism OID ::= { mechanism ttp(2) }
ttp-uni-mechanism OID ::= { ttp-mechanism uni(1) }
ttp-mut-mechanism OID ::= { ttp-mechanism mut(2) }
ttp-uni-1 OID ::= { ttp-uni-mechanism 1 }
ttp-uni-2 OID ::= { ttp-uni-mechanism 2 }
ttp-mut-1 OID ::= { ttp-mut-mechanism 1 }
ttp-mut-2 OID ::= { ttp-mut-mechanism 2 }
ttp-mut-3 OID ::= { ttp-mut-mechanism 3 }
END -- EntityAuthenticationMechanisms-3 –
```

A.2 Use of subsequent object identifiers

Immediately after an entity authentication mechanism identifier, an information object that identifies a digital signature algorithm [i.e. one of the algorithms specified in ISO/IEC 14888 (all parts) or ISO/IEC 9796 (all parts)] and any associated parameters (i.e. one of the hash functions specified in ISO/IEC 10118-3) shall follow.

Annex B
(informative)

Usage guidance

B.1 Security properties

B.1.1 Entity Authentication

The goal of entity authentication is to corroborate that an entity is the one claimed. To formalize this property, an attacker is considered that can perform man-in-the-middle, replay, reflection and forced delay attacks, according to ISO/IEC 9798-1. See Bibliographic entry [1] for an extensive security analysis of the second edition of this document. This report considers the following security properties:

— aliveness of the intended peer;

— weak agreement with the peer on the involved agents and their roles;

— data agreement with the peer on nonces and text fields, where possible.

Mechanisms that provide the above properties are resilient against man-in-the-middle attacks, reflection attacks and replay attacks (assuming proper usage of time variant parameters according to ISO/IEC 9798–1:2010, Annex B). For mechanisms involving a trusted third party, these properties are only considered between *A* and *B*, not between *TP* and *A*/*B*, although both entities *A* and *B* are assured of *TP*'s aliveness but not vice versa.

B.1.2 Unilateral and mutual authentication

A unilateral authentication protocol only guarantees the security properties with regard to one of the peers, i.e. entity *B* is assured of *A*'s aliveness, weak agreement, and data agreement but not vice versa. In mutual authentication, both peers are assured of each other's aliveness, weak agreement, and data agreement.

NOTE The requirement of weak and data agreement implies that mutual authentication cannot be achieved by the mere combination of two unilateral authentication mechanisms.

B.1.3 Certificate distribution and trust

All of the mechanisms allow the inclusion of I_X as an optional field in the messages. I_X contains either a certificate, $Cert_X$, or the identity, *X*. This allows the prover to provide information on its identity and/or certificate to the verifier. However, it does not provide the verifier with a way to determine its trust in the provided information. The verifier still needs to validate the certificate or, equivalently, needs to have a trusted copy of the public key of the prover. This validation step is however out of the scope of this document.

The mechanisms UNI.TS, UNI.CR, MUT.TS, MUT.CR and MUT.CR.par leave it up to the parties *A* and/or *B* to perform the validation of certificates and/or public keys (possibly involving a secondary mechanism such as OCSP with third parties involved).

The mechanisms TP.UNI.1, TP.UNI.2 and TP.MUT.1 to TP.MUT.3 directly include one or two trusted third parties (TP) to validate the public keys or certificates of the parties *A* and *B*. This transfers the task of validating certificates and/or public keys from the parties *A* and *B* to a *TP*. Note that it still remains out of the scope of this document how a *TP* performs this validation. It is assumed that the identity and public key of *TP* is known to both parties *A* and *B* (or in case of two TPs, each party knows the identity

and public key of the respective *TP*), i.e. the mechanisms can only be used in a closed environment with a fixed *TP*.

A *TP* interacts with only one of the parties during the execution of the authentication mechanism, but provides validation of certificates to both. A *TP* has to be available on-line for every execution of the mechanism. Moreover, a *TP* learns the unauthenticated identities of both parties involved at each execution of the mechanism. In some setups it is undesirable that a third party gets information on the identity of the parties communicating, regardless of this information being authentic or not.

B.2 Comparison and selection of mechanisms

B.2.1 Comparison

Table B.1 gives an overview of the security properties of the mechanisms specified in this document and the known limitations of the protocols. The table only shows the properties when all optional fields are included. Omitting these fields can result in lower security guarantees, as detailed in the notes for the respective mechanisms. The security issues presented in Bibliographic entry [1] have been fixed (to the extent possible) by tagging each signed message with a unique identifier and by ensuring a unique decoding of concatenated strings throughout the mechanisms.

Table B.1 — Security properties of mechanisms

	Mutual	TP	Messages	Freshness or uniqueness by
Mechanism UNI.TS	N	N	1	T_A / N_A
Mechanism UNI.CR	N	N	2	R_B
Mechanism MUT.TS	Y	N	2	T_A / N_A and T_B / N_B
Mechanism MUT.CR	Y	N	3	R_A and R_B
Mechanism MUT.CR.par	Y	N	4	R_A and R_B
Mechanism TP.UNI.1	N	Y	4	R_A and R'_A
Mechanism TP.UNI.2	N	Y	4	R_B
Mechanism TP.MUT.1	Y	Y	5	R_A, R'_A and R_B
Mechanism TP.MUT.2	Y	Y	5	R_A, R'_A and R_B
Mechanism TP.MUT.3	Y	Y	7	R_A, R'_A and R_B

B.2.2 Recommendations for the selection of a mechanism

Several criteria should be taken into account when selecting an appropriate mechanism for authentication:

— The need for a secure communication channel: the mechanisms in this document do not provide any confidentiality of the communication and moreover do not set up a secure channel for further communication after authentication is finished. If a secure communication channel is required, a key establishment method from ISO/IEC 11770[6] should be used instead of the mechanisms from this document.

— Unilateral or mutual authentication: the need for unilateral or mutual authentication is completely determined by the application.

— Known security limitations: the mechanisms UNI.1 and MUT.1 ensure freshness/uniqueness of the authentication by using a time stamp or sequence numbers. As detailed in ISO/IEC 9798–1:2010, Annex B, this requires synchronized clocks (for time stamps) or additional book keeping for validating sequence numbers. The absence of these additional measures will open the mechanisms to attacks. In many cases, a challenge-response mechanism is preferred to avoid these additional requirements.

— Communication and computation complexity: the efficiency of the mechanisms is determined by the requirements for communication (number, size of messages) and computation (generation and validation of signatures and random number generation) by each of the parties. These costs are highly dependent on the platform that is used for the implementation of a mechanism: possible influencing factors include the throughput and delay of the network, whether uni- or bi-directional communication is in use, the capabilities of the processor and memory of both parties, etc.

— Certificate and key validation: how validation is achieved is outside the scope of this document. All the mechanisms can be amended with a certificate validation mechanism that each of the parties can execute (either directly or indirectly through the other party). In closed environments, where a single entity can be made responsible for validating certificates/keys, use of one of the TP.*-mechanisms can be considered.

Annex C
(informative)

Use of text fields

The tokens specified in Clauses 7 and 8 contain text fields. The actual use of and the relationships between the various text fields in a given pass depend on the application. Some examples are given below; see also ISO/IEC 9798-1:2010, Annex A.

If a signature scheme without message recovery is used and if the signed text field is not empty, then the verifier needs to be in possession of the text prior to verifying the signature. In Annex C, "signed text fields" refers to text fields in the signed data and "unsigned text fields" refers to text fields in the unsigned data.

For example, if a digital signature scheme without message recovery is used, any information requiring data origin authentication should be placed in the signed text field and (as part of) the unsigned text field in the token.

If the tokens do not contain (sufficient) redundancy, the signed text fields may be used to provide additional redundancy.

Signed text fields may be used to indicate that the token is only valid for the purpose of entity authentication. Should there be a concern that one entity can choose a "degenerate" value with malicious intent for the other entity to sign, the other entity may introduce a random number in the text field.

Should an algorithm be used where it can be possible to launch attacks based on the fact that a particular claimant is using the same key for all verifiers with which the claimant communicates, and if such attacks are considered to be a threat, the identity of the intended verifier should be included in the signed text field and, if necessary, in the unsigned text field.

Unsigned text fields can also be used to provide information to a verifier indicating the (unauthenticated) identity which a claimant is claiming. If means other than certificates are used for distributing public keys, such information can be required to allow a verifier to determine which public key is to be used to authenticate a claimant.

Bibliography

[1] Basin D., & Cremers C. Evaluation of ISO/IEC 9798 Protocols *CRYPTREC Technical Report, Version 2.0, April* 2011. Available at https://www.cryptrec.go.jp/estimation/techrep_id2014_2.pdf

[2] ISO/IEC 8824 (all parts), *Information technology — Abstract Syntax Notation One (ASN.1)*

[3] ISO/IEC 8825 (all parts), *Information technology — ASN.1 Encoding rules*

[4] ISO/IEC 9594-8, *Information technology — Open Systems Interconnection — The Directory — Part 8: Public-key and attribute certificate frameworks*

[5] ISO/IEC 10118-3, *IT Security techniques — Hash-functions — Part 3: Dedicated hash-functions*

[6] ISO/IEC 11770 (all parts), *Information technology — Security techniques — Key management*

[7] ITU-T X.509, *Information technology — Open systems interconnection — The directory: public-key and attribute certificate frameworks*

British Standards Institution (BSI)

BSI is the national body responsible for preparing British Standards and other standards-related publications, information and services.

BSI is incorporated by Royal Charter. British Standards and other standardization products are published by BSI Standards Limited.

About us

We bring together business, industry, government, consumers, innovators and others to shape their combined experience and expertise into standards -based solutions.

The knowledge embodied in our standards has been carefully assembled in a dependable format and refined through our open consultation process. Organizations of all sizes and across all sectors choose standards to help them achieve their goals.

Information on standards

We can provide you with the knowledge that your organization needs to succeed. Find out more about British Standards by visiting our website at bsigroup.com/standards or contacting our Customer Services team or Knowledge Centre.

Buying standards

You can buy and download PDF versions of BSI publications, including British and adopted European and international standards, through our website at bsigroup.com/shop, where hard copies can also be purchased.

If you need international and foreign standards from other Standards Development Organizations, hard copies can be ordered from our Customer Services team.

Copyright in BSI publications

All the content in BSI publications, including British Standards, is the property of and copyrighted by BSI or some person or entity that owns copyright in the information used (such as the international standardization bodies) and has formally licensed such information to BSI for commercial publication and use.

Save for the provisions below, you may not transfer, share or disseminate any portion of the standard to any other person. You may not adapt, distribute, commercially exploit, or publicly display the standard or any portion thereof in any manner whatsoever without BSI's prior written consent.

Storing and using standards

Standards purchased in soft copy format:

- A British Standard purchased in soft copy format is licensed to a sole named user for personal or internal company use only.
- The standard may be stored on more than 1 device provided that it is accessible by the sole named user only and that only 1 copy is accessed at any one time.
- A single paper copy may be printed for personal or internal company use only.
- Standards purchased in hard copy format:
- A British Standard purchased in hard copy format is for personal or internal company use only.
- It may not be further reproduced – in any format – to create an additional copy. This includes scanning of the document.

If you need more than 1 copy of the document, or if you wish to share the document on an internal network, you can save money by choosing a subscription product (see 'Subscriptions').

Reproducing extracts

For permission to reproduce content from BSI publications contact the BSI Copyright & Licensing team.

Subscriptions

Our range of subscription services are designed to make using standards easier for you. For further information on our subscription products go to bsigroup.com/subscriptions.

With **British Standards Online (BSOL)** you'll have instant access to over 55,000 British and adopted European and international standards from your desktop. It's available 24/7 and is refreshed daily so you'll always be up to date.

You can keep in touch with standards developments and receive substantial discounts on the purchase price of standards, both in single copy and subscription format, by becoming a **BSI Subscribing Member**.

PLUS is an updating service exclusive to BSI Subscribing Members. You will automatically receive the latest hard copy of your standards when they're revised or replaced.

To find out more about becoming a BSI Subscribing Member and the benefits of membership, please visit bsigroup.com/shop.

With a **Multi-User Network Licence (MUNL)** you are able to host standards publications on your intranet. Licences can cover as few or as many users as you wish. With updates supplied as soon as they're available, you can be sure your documentation is current. For further information, email subscriptions@bsigroup.com.

Revisions

Our British Standards and other publications are updated by amendment or revision.

We continually improve the quality of our products and services to benefit your business. If you find an inaccuracy or ambiguity within a British Standard or other BSI publication please inform the Knowledge Centre.

Useful Contacts

Customer Services
Tel: +44 345 086 9001
Email (orders): orders@bsigroup.com
Email (enquiries): cservices@bsigroup.com

Subscriptions
Tel: +44 345 086 9001
Email: subscriptions@bsigroup.com

Knowledge Centre
Tel: +44 20 8996 7004
Email: knowledgecentre@bsigroup.com

Copyright & Licensing
Tel: +44 20 8996 7070
Email: copyright@bsigroup.com

BSI Group Headquarters

389 Chiswick High Road London W4 4AL UK

bsi.