INTERNATIONAL STANDARD

**ISO/IEC 9798-5**

First edition
1999-03-15

# Information technology — Security techniques — Entity authentication —

## Part 5:
## Mechanisms using zero knowledge techniques

*Technologies de l'information — Techniques de sécurité — Authentification d'entité —*

*Partie 5: Mécanismes utilisant les techniques de connaissance du zéro*

Reference number
ISO/IEC 9798-5:1999(E)

## Contents

## 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 JTC1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

International Standard ISO/IEC 9798–5 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Sub-Committee SC27, *IT Security techniques*.

ISO/IEC 9798 consists of the following parts, under the general title *Information technology — Security techniques — Entity authentication*:

— *Part 1: General*

— *Part 2: Mechanisms using symmetric encipherment algorithms*

— *Part 3: Mechanisms using digital signature techniques*

— *Part 4: Mechanisms using a cryptographic check function*

— *Part 5: Mechanisms using zero knowledge techniques*

Annexes A, B, C, D, E, and F of this part of ISO/IEC 9798 are for information only.

ISO and IEC draw attention to the fact that it is claimed that compliance with this part of ISO/IEC 9798 may involve the use of patents as given in Annex E.

ISO and IEC take no position regarding the evidence, validity and scope of these patent rights.

The holders of these patent rights have assured ISO and IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of these patent rights are registered with ISO and IEC. Information may be obtained from the addresses given in Annex E.

Attention is drawn to the possibility that some of the elements of this part of ISO/IEC 9798 may be the subject of patent rights other than those identified in Annex E. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

# Information technology — Security techniques — Entity authentication — Part 5: Mechanisms using zero knowledge techniques

## 1 Scope

This part of ISO/IEC 9798 specifies three entity authentication mechanisms using zero knowledge techniques. All the mechanisms specified in this part of ISO/IEC 9798 provide unilateral authentication. These mechanisms are constructed using the principles of zero knowledge, but they will not be zero knowledge according to the strict definition sketched in Annex A for all choices of parameters.

The first mechanism is said to be based on identities. A trusted accreditation authority provides each claimant with private accreditation information, computed as a function of the claimant's identification data and the accreditation authority's private key.

The second mechanism is said to be certificate-based using discrete logarithms. Every claimant possesses a public key, private key pair for use in this mechanism. Every verifier of a claimant's identity must possess a trusted copy of the claimant's public verification key; the means by which this is achieved is beyond the scope of this standard, but it may be achieved through the distribution of certificates signed by a Trusted Third Party.

The third mechanism is said to be certificate-based using an asymmetric encipherment system. Every claimant possesses a public key, private key pair for an asymmetric cryptosystem. Every verifier of a claimant's identity must possess a trusted copy of the claimant's public key; the means by which this is achieved is beyond the scope of this standard, but it may be achieved through the distribution of certificates signed by a Trusted Third Party.

## 2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 9798. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this International Standard are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO/IEC 9796: 1991, *Information technology — Security techniques — Digital signature scheme giving message recovery.*

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

ISO/IEC 10118 (all parts), *Information technology — Security techniques — Hash-functions.*

## 3 Definitions

For the purposes of this part of ISO/IEC 9798, the following definitions apply.

The following terms are defined in ISO/IEC 9798-1.

**3.1 asymmetric cryptographic technique:**

**3.2 asymmetric encipherment system:**

**3.3 asymmetric key pair:**

**3.4 challenge:**

**3.5 claimant:**

**3.6 decipherment:**

**3.7 distinguishing identifier:**

**3.8 encipherment:**

**3.9 entity authentication:**

**3.10 private key:**

**3.11 public key:**

**3.12 public verification key:**

**3.13 random number:**

**3.14 token:**

**3.15 trusted third party:**

**3.16 unilateral authentication:**

**3.17 verifier:**

The following term is defined in ISO/IEC 10118-1.

**3.18 hash-function:** function which maps strings of bits to fixed-length strings of bits, satisfying the following two properties:

- it is computationally infeasible to find for a given output an input which maps to this output;
- it is computationally infeasible to find for a given input a second input which maps to the same output.

In addition the following definitions are used.

**3.19 accreditation authority:** entity trusted by all members of a group of entities for the purposes of the generation of private accreditation information.

**3.20 accreditation multiplicity parameter:** positive integer equal to the number of items of secret accreditation information provided to an entity by the accreditation authority.

**3.21 exchange multiplicity parameter:** positive integer used to determine how many times the exchange of entity authentication messages shall be performed in one instance of the authentication mechanism.

**3.22 identification data:** sequence of data items, including the distinguishing identifier for an entity, assigned to an entity and used to identify it.

NOTE — Examples of data items which may be included in the identification data include: an account number, expiry date, serial number, etc.

**3.23 private accreditation exponent:** value known only to the accreditation authority, and which is used in the production of claimants' private accreditation information. This value shall be kept secret. This value is related to the public accreditation verification exponent.

**3.24 private accreditation information:** private information provided to a claimant by an accreditation authority, and of which a claimant subsequently proves knowledge, thereby establishing the claimant's identity.

**3.25 private decipherment transformation:** decipherment transformation determined by an asymmetric encipherment system and the private key of an asymmetric key pair.

**3.26 public accreditation verification exponent:** value agreed by all members of a group of entities, and which, in conjunction with the modulus, determines the value of the private accreditation exponent.

**3.27 public encipherment transformation:** encipherment transformation determined by an asymmetric encipherment system and the public key of an asymmetric key pair.

**3.28 redundant identity:** sequence of data items obtained from an entity's identification data by adding redundancy using techniques specified in ISO/IEC 9796.

**3.29 response:** data item sent by the claimant to the verifier, and which the verifier can process to help check the identity of the claimant.

**3.30 witness:** data item which provides evidence of the claimant's identity to the verifier.

## 4 Symbols and notation

For the purposes of this part of ISO/IEC 9798 the following symbols and notation described in ISO/IEC 9798-1 apply.

$A$ — The distinguishing identifier of entity $A$.

$B$ — The distinguishing identifier of entity $B$.

$Y||Z$ — The result of the concatenation of the data items $Y$ and $Z$ in that order.

The following general symbols and notation are used.

$d$ — A challenge.

$D$ — A response.

$h$ — A hash-function.

$r$ — A random number.

$\lfloor x \rfloor$ — The largest integer which is not greater than the value $x$.

mod — If $i$ is an integer and $n$ is a positive integer, then $i \bmod n$ denotes the unique integer $j$ which satisfies

a) $0 \leq j < n$, and

b) $i - j$ is an integer multiple of $n$.

In the context of the identity-based mechanism of clause 5, the following symbols and notation are used.

$C_{A1}, C_{A2}, \ldots, C_{Am}$ — Entity $A$'s private accreditation information.

gcd — The greatest common divisor of two integers, i.e. $\gcd(a, b)$ represents the largest positive integer that is a divisor of both $a$ and $b$.

$I_{A1}, I_{A2}, \ldots, I_{Am}$ — The identification data of entity $A$. $I_{Ai}$ is the $i$th part of the identification data of entity $A$.

$J_{A1}, J_{A2}, \ldots, J_{Am}$ — The redundant identity of entity $A$. $J_{Ai}$ is the $i$th part of the redundant identity of entity $A$.

$k_s$ — An integer determined by the modulus $n$, and which determines the maximum bit-length of the parts of an entity's redundant identity.

lcm — The least common multiple of two integers, i.e. $\mathrm{lcm}(a, b)$ represents the smallest positive integer that is a multiple of both $a$ and $b$.

$m$ — The accreditation multiplicity parameter.

$n$ — A modulus equal to the product of the prime numbers $p$ and $q$.

$p$ — A prime number used to calculate the modulus.

$q$ — A prime number used to calculate the modulus.

$t$ — The exchange multiplicity parameter.

$u$ — The accreditation authority's private accreditation exponent.

$v$ — The public accreditation verification exponent.

$W$ — A witness.

mod* — If $i$ is an integer and $n$ is a positive integer, then $i \bmod^* n$ denotes the non-negative integer $j$ equal to the smaller of the two values: $i \bmod n$ and $n - i \bmod n$. If $i$ and $j$ are two integers and $n$ is a positive integer then $i \equiv j \pmod^* n$ if and only if $i \bmod^* n = j \bmod^* n$.

$(a|n)$ — The Jacobi symbol of the positive integer $a$ with respect to the odd positive integer $n$.

NOTE — Let $p$ be an odd prime and let $a$ be a positive integer. The Legendre symbol of $a$ with respect to $p$, written $(a|p)$, is defined by

$$(a|p) = a^{(p-1)/2} \bmod p.$$

When $a$ is not a multiple of $p$, $(a|p)$ is either +1 or -1, depending on whether or not $a$ is equal to the square of an integer modulo $p$. The Legendre symbol of multiples of $p$ with respect to a prime $p$ is zero.

Let $n$ be an odd positive integer satisfying $n = pq$, where $p$ and $q$ are primes, and let $a$ be a positive integer. The Jacobi symbol of $a$ with respect to $n$, written $(a|n)$, is defined by

$$(a|n) = (a|p)(a|q).$$

The Jacobi symbol $(a|n)$ can be computed efficiently without knowledge of the prime factorisation of $n$, see [6] and [8].

In the context of the discrete logarithm based mechanism of clause 6, the following symbols and notation are used.

$g$ — A positive integer which is the base of the discrete logarithms.

$p$ — A prime number used as a modulus.

$q$ — A prime number which is a factor of $p - 1$.

$y_X$ — The public verification key for entity $X$.

$z_X$ — The private authentication key for entity $X$.

In the context of the trusted public transformation based mechanism of clause 7, the following symbols and notation are used.

$P_X$ — The public encipherment transformation for entity $X$.

$S_X$ — The private decipherment transformation for entity $X$.

## 5 Mechanism based on identities

In this clause an entity authentication mechanism is specified which is based on the use of identities.

### 5.1 Specific requirements

In order to use the mechanism within a group of entities, the following steps shall be taken.

a) Every entity wishing to act as either a claimant or a verifier must have the means to generate random numbers.

b) An accreditation authority shall be appointed for the group of entities. This accreditation authority shall be trusted by all members of the group for the purposes of guaranteeing identities.

c) A number of parameters shall be selected, which will govern the operation of the entity authentication mechanism. The selected parameters shall be made known in a reliable manner to all members of the group of entities.

d) Every entity wishing to act as a claimant in the authentication mechanism must be provided with identification data by some means. In this context identification data is a string of bits of length limited by one of the parameters selected in step (c), and which uniquely and meaningfully identifies the entity according to an agreed convention.

e) Every entity wishing to act as a claimant in the authentication mechanism shall be issued with private accreditation information by the selected accreditation authority.

f) If the version of the mechanism using a hash-function is selected, all entities within the group must agree on the use of a specific hash-function (for example one of the functions specified in ISO/IEC 10118).

## 5.2 Parameter selection

The parameters to be selected are as follows.

a) The public accreditation verification exponent $v$. Certain values, such as 2, 3 and $2^{16}+1 = 65537$, have some practical advantages.

b) The modulus $n$. This positive integer shall be selected by the appointed accreditation authority. The value of $n$ shall be equal to the product of two prime numbers $p$ and $q$. The values of $p$ and $q$ shall be kept secret by the accreditation authority. The prime numbers $p$ and $q$ shall be chosen in such a way that knowledge of their product $n$ shall not feasibly enable any entity to deduce them, where feasibility is defined by the context of use of the authentication mechanism.

The values of $p$ and $q$ shall satisfy the following constraints:

— if $v$ is odd, then $\gcd(p-1, v) = \gcd(q-1, v) = 1$, and

— if $v$ is even, then $\gcd((p-1)/2, v) = \gcd((q-1)/2, v) = 1$, and $p-q$ shall not be a multiple of 8.

The choice of $n$ determines the value of a further parameter, which is denoted by $k_s$, in the following way:

$$k_s = \lfloor \log_2(n) \rfloor.$$

In other words, a binary representation of $n$ shall contain $k_s + 1$ bits.

The choice of $n$ also determines the value of the accreditation authority's private accreditation exponent, denoted $u$, in the following way. The value $u$ shall be set to the least positive integer such that $uv+1$ is a multiple of

$$\begin{array}{ll} \mathrm{lcm}(p-1, q-1) & \text{if } v \text{ is odd,} \\ \mathrm{lcm}(p-1, q-1)/2 & \text{if } v \text{ is even.} \end{array}$$

c) The accreditation multiplicity parameter $m$. This positive integer shall be chosen in conjunction with the public accreditation verification exponent $v$ and the exchange multiplicity $t$, and affects the level of security of the scheme.

d) The exchange multiplicity parameter $t$. This positive integer shall be chosen in conjunction with the public accreditation verification exponent $v$ and the accreditation multiplicity $m$, and affects the level of security of the scheme.

NOTE 1 — Guidance on the choice of parameters for this mechanism is given in Annex B.

NOTE 2 — When $v = 2$ the mechanism becomes the Fiat-Shamir scheme, [3]. When $v > 2$, $m = 1$, and $v$ is a prime the mechanism becomes the Guillou-Quisquater scheme, [5].

## 5.3 Identity selection

Each entity wishing to act as a claimant in this mechanism must be assigned identification data consisting of a sequence of $m$ parts: $I_{A1}, I_{A2}, \ldots, I_{Am}$. Each part of the identification data shall contain at most $8\lfloor(k_s+3)/16\rfloor$ bits.

The entity authentication mechanism will provide assurance to the verifier that the claimant is the entity which has been assigned this identification data.

NOTE 1 — This sequence of identification data parts could, for example, be constructed by assigning an entity a single identifying string of bits, and appending the binary representations of the numbers $1, 2, \ldots, m$ in turn to this string to obtain the values $I_{A1}, I_{A2}, \ldots, I_{Am}$. In such an approach, the binary representations of the numbers $1, 2, \ldots, m$ could conveniently all be made of the same length, by prefixing with zeros as necessary.

NOTE 2 — If the parts of an entity's identification data are longer than the maximum permissible length, then this can be dealt with by applying a hash-function to the identification data parts to obtain the values $I_{A1}, I_{A2}, \ldots, I_{Am}$. Examples of hash-functions can be found in ISO/IEC 10118.

NOTE 3 — Expiry of an entity's identification data can be enforced by the inclusion of an expiry date in the identification data. Revocation of an entity's identification data can be simplified by the inclusion of a serial number in the identification data.

## 5.4 Accreditation generation

To generate the private accreditation information for an entity $A$, the accreditation authority shall compute a sequence of $m$ digital signatures $C_{A1}, C_{A2}, \ldots, C_{Am}$. More specifically, for every $i$ $(1 \le i \le m)$, $C_{Ai}$ shall be computed using the following procedure.

a) $J_{Ai}$, the $i$th part of the 'redundant identity' for $A$, shall be computed from $I_{Ai}$, the $i$th part of the identification data of $A$, by subjecting $I_{Ai}$ to the first four steps of the signature process specified in ISO/IEC 9796, ('Padding', 'Extension', 'Redundancy' and 'Truncation and forcing'), using the specified value of $k_s$. The value obtained from this process, denoted $IR$, shall then be used to derive $J_{Ai}$ in the following way.

— If $v$ is odd then $J_{Ai} = IR$.

— If $v$ is even and if $(IR|n) = +1$ then $J_{Ai} = IR$.

— If $v$ is even and if $(IR|n) = -1$ then $J_{Ai} = IR/2$.

b) $C_{Ai}$ shall be computed from $J_{Ai}$ using the following formula:

$$C_{Ai} = (J_{Ai})^u \bmod^* n.$$

The private accreditation information supplied to entity $A$ is equal to the computed signatures $C_{A1}, C_{A2}, \ldots, C_{Am}$.

Observe that

$$(C_{Ai})^v J_{Ai} \equiv 1 \pmod^* n)$$

for every $i$ $(1 \le i \le m)$.

## 5.5 Authentication exchange

This unilateral authentication mechanism involves the following exchanges of information between a claimant $A$ and a verifier $B$, and enables $B$ to check the identity of $A$. It is necessary for correct operation of the mechanism that $B$ is provided with the claimed identification data of $A$, either appended to one of the information exchanges in the mechanism or by some other means.

One iteration of the authentication procedure is illustrated in figure 1. The bracketed numbers in the figure correspond to the steps of the exchange described in detail below.

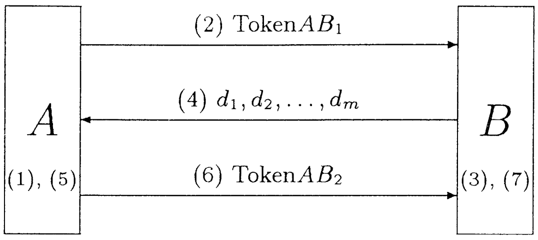

**Figure 1 — Identity-based mechanism**

The form of the first token (Token$AB_1$), sent by the claimant to the verifier is either:

$$\text{Token}AB_1 = W$$

or

$$\text{Token}AB_1 = h(W||\text{Text})$$

where $W$ is the witness, $h$ is a hash-function, and Text is an optional text field. This text field is available for use in applications outside the scope of this part of ISO/IEC 9798 (it may be empty). See annex A of ISO/IEC 9798-1 for information on the use of text fields. If this text field is non-empty then $B$ must have the means to recover the value of the text field; this may require $A$ to send all or part of the text field with Token$AB_1$ (see also Note 1 below).

The form of the second token (Token$AB_2$), sent by the claimant to the verifier is:

$$\text{Token}AB_2 = D$$

where $D$ is the response.

For each application of this mechanism the following authentication procedure shall be performed $t$ times (where $t$ is the exchange multiplicity parameter). The verifier $B$ shall only accept the claimant $A$ as valid if all $t$ iterations of the procedure complete successfully.

(1) Entity $A$, who is equipped with private accreditation information $C_{A1}, C_{A2}, \ldots, C_{Am}$, chooses a random number $r$, subject to the restriction that $r$ shall be an integer satisfying $1 \leq r \leq n-1$. This integer is kept secret by $A$. $A$ now computes the witness $W$ as

$$W = r^v \bmod^* n.$$

(2) $A$ sends Token$AB_1$ to $B$. Token$AB_1$ shall be equal to either $W$ or $h(W||\text{Text})$.

(3) Having received Token$AB_1$, $B$ shall choose at random a sequence of integers $d_1, d_2, \ldots, d_m$, where each value $d_i$ shall lie in the range 0 to $v-1$. This sequence of integers is the challenge.

(4) $B$ sends the challenge $d_1, d_2, \ldots, d_m$ to $A$.

(5) On receipt of the challenge $d_1, d_2, \ldots, d_m$, $A$ shall compute the response $D$ from the (secret) value $r$ and the private accreditation information $C_{A1}, C_{A2}, \ldots, C_{Am}$ as follows:

$$D = r \prod_{i=1}^{m} (C_{Ai})^{d_i} \bmod^* n.$$

(6) $A$ sends Token$AB_2 = D$ to $B$.

(7) On receipt of the response $D$, $B$ shall perform the following computations.

(a) $B$ checks that $0 < D < n/2$. If not then $B$ shall reject $A$.

(b) $B$ calculates $J_{A1}, J_{A2}, \ldots, J_{Am}$, the redundant identity of $A$, from the identification data $I_{A1}, I_{A2}, \ldots, I_{Am}$ of $A$, using the same process as specified in clause 5.4, step (a).

(c) $B$ now computes the value $W'$ using the following formula:

$$W' = D^v \prod_{i=1}^{m} (J_{Ai})^{d_i} \bmod^* n.$$

(d) If $W$ was sent in the first exchange of the procedure then $B$ checks that the computed value $W'$ is equal to the value of $W$ sent in the first exchange of the procedure. Alternatively, if $h(W||\text{Text})$ was sent in the first exchange of the procedure then $B$ first computes $h(W'||\text{Text})$ and then checks that $h(W'||\text{Text})$ is equal to the value of $h(W||\text{Text})$ sent in the first exchange of the procedure. If the check succeeds then this iteration of the mechanism is successful. Otherwise $B$ rejects $A$.

NOTE 1 — Other information may be sent with any of the exchanges of any of the iterations of the authentication procedure. In particular note that information included within the identification data of $A$ may be sent with Token$AB_1$ in the first exchange of the first of the $t$ iterations of the authentication procedure (step (2)). Such information might be used by $B$ to help compute $A$'s redundant identity and/or the value of the optional Text field.

NOTE 2 — It is important that $A$ chooses the random value $r$ by a process which guarantees that selected values are independent within the lifetime of the accreditation information. If, for example, the same value $r$ is used twice, then a third party may be able to deduce part or all of the private accreditation information of $A$, and hence be able to impersonate $A$ successfully.

NOTE 3 — The redundant identity $J_{A1}, J_{A2}, \ldots, J_{Am}$ for $A$ can be computed at any stage by $B$, i.e. $B$ need not wait until the receipt of the response $D$ before computing $J_{A1}, J_{A2}, \ldots, J_{Am}$. If $B$ verifies $A$ frequently using this mechanism, then $B$ may cache the values $J_{A1}, J_{A2}, \ldots, J_{Am}$.

NOTE 4 — The $t$ iterations of the procedure can be performed in parallel, i.e. in the first step $A$ may choose $t$ random numbers $r_1, r_2, \ldots, r_t$, compute $t$ witnesses $W_1, W_2, \ldots, W_t$, send them simultaneously to $B$, and so on. If this 'parallel implementation' is adopted, the total number of message exchanges will be equal to three, regardless of the value of $t$.

NOTE 5 — The use of $h(W||\text{Text})$ instead of $W$ in the first exchange of the procedure can achieve efficiency gains by reducing the number of bits in Token$AB_1$.

NOTE 6 — It is recommended that the private accreditation information used in this mechanism be used only for the purposes of authentication, and should not be used for any other application (e.g. generation of digital signatures). If this recommendation is not followed then special care should be taken to prevent the verifier using the claimant as a 'signing oracle'; this could, for example, be achieved by requiring the challenge to be of a specially chosen form.

## 6 Certificate-based mechanism using discrete logarithms

In this clause an entity authentication mechanism is specified which uses discrete logarithms.

NOTE — This mechanism is known as the Schnorr scheme, [10].

### 6.1 Specific requirements

In order to use the mechanism within a group of entities, the following steps shall be taken.

a) Every entity wishing to act as either a claimant or a verifier must have the means to generate random numbers.

b) All entities within the group must agree on three positive integers $p$, $q$ and $g$. The integer $p$ must be chosen to be a prime number. Further $q$ must be chosen to be a prime number which is also a factor of $p-1$. Finally $g$ must be chosen to be an element of order $q$ modulo $p$, that is $g$ must satisfy:

(i) $g^q \bmod p = 1$, and

(ii) $g \neq 1$.

The values $p$ and $g$ should be chosen so that, given an arbitrary integer $i$ $(1 < i < q)$, finding an integer $j$ (if one exists) such that $g^j \bmod p = i$ shall be computationally infeasible.

c) All entities within the group must agree on the use of a hash-function (for example one of the functions specified in ISO/IEC 10118).

d) Every entity wishing to act as a claimant must be equipped with an asymmetric key pair, selected as described below.

e) Every entity wishing to act as a verifier must be equipped with a means to obtain trusted copies of the public verification keys for the entities whose identities it is to verify.

NOTE — The exact means by which entities are provided with trusted copies of public verification keys is beyond the scope of this standard. This may, for example, be achieved by the use of public key certificates or by some other environment-dependent means.

### 6.2 Key selection

Each entity $X$ wishing to act as a claimant in this mechanism must be equipped with an asymmetric key pair $(y_X, z_X)$, where $z_X$ (the private key) shall be an integer chosen to satisfy $0 < z_X < q$. The corresponding public verification key $y_X$ shall be set equal to $g^{z_X} \bmod p$.

NOTE — Guidance on the choice of parameters for this mechanism is given in Annex B.

### 6.3 Authentication exchange

This unilateral authentication mechanism involves the following exchanges of information between a claimant $A$ and a verifier $B$, and enables $B$ to check the identity of $A$.

The authentication mechanism is illustrated in figure 2. The bracketed numbers in the figure correspond to the steps of the exchange described in detail below.

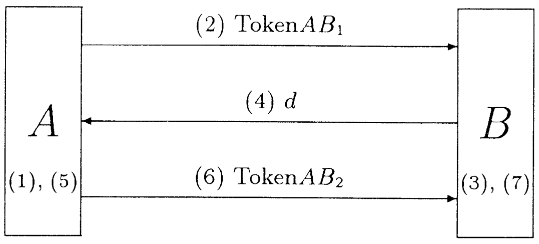

**Figure 2 — Discrete logarithm based mechanism**

The form of the first token (Token$AB_1$), sent by the claimant to the verifier is either:

$$\text{Token}AB_1 = W$$

or

$$\text{Token}AB_1 = h(W||\text{Text})$$

where $W$ is the witness, $h$ is a hash-function, and Text is an optional text field. This text field is available for use in applications outside the scope of this part of ISO/IEC 9798 (it may be empty). See annex A of ISO/IEC 9798-1 for information on the use of text fields. If this Text field is non-empty then $B$ must have the means to recover the value of Text; this may require $A$ to send all or part of the Text field with Token$AB_1$ (see also Note 1 below).

The form of the second token (Token$AB_2$), sent by the claimant to the verifier is:

$$\text{Token}AB_2 = D$$

where $D$ is the response.

(1) Entity $A$ chooses a random number $r$, subject to the restriction that $r$ shall be an integer satisfying $1 < r < q$. This integer is kept secret by $A$. $A$ now computes the witness $W$ as

$$W = g^r \bmod p.$$

(2) $A$ sends Token$AB_1$ to $B$. Token$AB_1$ shall be equal to either $W$ or $h(W||\text{Text})$.

(3) Having received Token$AB_1$, $B$ shall choose at random an integer $d$ (the 'challenge'), where $d$ shall satisfy $0 \leq d < q$.

(4) $B$ sends the challenge $d$ to $A$.

(5) On receipt of the challenge $d$, $A$ shall compute the response $D$ from the (secret) value $r$ and $A$'s private key $z_A$ by:

$$D = r - dz_A \bmod q.$$

(6) $A$ sends Token$AB_2 = D$ to $B$.

(7) On receipt of the response $D$, $B$ shall perform the following computations.

(a) $B$ checks that $0 < D < q$. If not then $B$ shall reject $A$.

(b) $B$ computes the value $W'$ using the following formula:

$$W' = (y_A)^d g^D \bmod p.$$

(c) If $W$ was sent in the first exchange of the procedure then $B$ checks that the computed value $W'$ is equal to the value of $W$ sent in the first exchange of the procedure. Alternatively, if $h(W||\text{Text})$ was sent in the first exchange of the procedure, then $B$ computes $h(W'||\text{Text})$ and then checks it is equal to the value of $h(W||\text{Text})$ sent in the first exchange of the procedure. If $h(W||\text{Text}) \neq h(W'||\text{Text})$ then the mechanism has failed and $A$ shall be rejected. Otherwise $B$ accepts $A$.

NOTE 1 — Other information may be sent with any of the exchanges of any of the iterations of the authentication procedure.

NOTE 2 — It is important that $A$ chooses the random value $r$ by a process which guarantees that selected values are independent within the lifetime of the accreditation information. If, for example, the same value $r$ is used twice, then a third party may be able to deduce the private accreditation information of $A$, and hence be able to impersonate $A$ successfully.

NOTE 3 — It is recommended that the key pair used in this mechanism be used only for the purposes of authentication, and should not be used for any other application (e.g. generation of digital signatures). If this recommendation is not followed then special care should be taken to prevent the verifier using the claimant as a 'signing oracle'; this could, for example, be achieved by requiring the challenge to be of a specially chosen form.

NOTE 4 — The use of $h(W||\text{Text})$ instead of $W$ in the first exchange of the procedure can achieve efficiency gains by reducing the number of bits in Token$AB_1$.

## 7 Certificate-based mechanism using an asymmetric encipherment system

In this clause an entity authentication mechanism is specified which uses an asymmetric encipherment system.

NOTE — This mechanism is derived from the Brandt-Damgard-Landrock-Pedersen scheme, [1,8].

### 7.1 Specific requirements

In order to use the mechanism within a group of entities, the following steps shall be taken.

a) Every entity wishing to act as a verifier must have the means to generate random numbers.

b) All entities within the group must agree on the use of two cryptographic functions: an asymmetric encipherment system, and a hash-function (for example one of the functions specified in ISO/IEC 10118).

c) Every entity wishing to act as a claimant must be equipped with an asymmetric key pair for use with the asymmetric encipherment system.

d) Every entity wishing to act as a verifier must be equipped with a means to obtain trusted copies of the public keys for the entities whose identities it is to verify.

NOTE — The exact means by which entities are provided with trusted copies of public keys is beyond the scope of this standard. This may, for example, be achieved by the use of public key certificates or by some other environment-dependent means.

### 7.2 Authentication exchange

This unilateral authentication mechanism involves the following exchanges of information between a claimant $A$ and a verifier $B$, and enables $B$ to check the identity of $A$.

The authentication mechanism is illustrated in figure 3. The bracketed numbers in the figure correspond to the steps of the exchange described in detail below.

The form of the token (Token$BA$) sent by the verifier to the claimant is:

$$\text{Token}BA = d$$

where $d$ is the challenge. The form of the token (Token$AB$) sent by the claimant to the verifier is:

$$\text{Token}AB = D$$

where $D$ is the response.

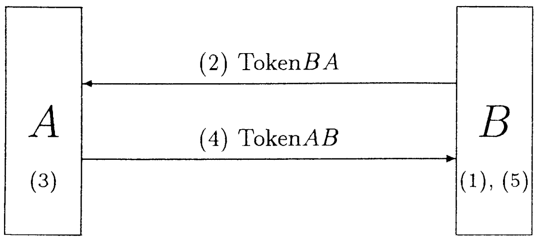

**Figure 3 — Trusted public transformation based mechanism**

(1) Entity $B$ chooses a random number $r$. This number is kept secret by $B$. $B$ next computes $h(r)$. The random number $r$ shall be chosen in such a way that $r||h(r)$ lies within the domain of $P_A$, the public encipherment transformation of $A$. $B$ now computes the challenge $d$ as

$$d = P_A(r||h(r)).$$

(2) $B$ sends Token$BA = d$ to $A$.

(3) Having received Token$BA$, $A$ performs the following computational steps.

(a) $A$ recovers the value $r$ by calculating

$$r||h(r) = S_A(d),$$

where $S_A$ is the private decipherment transformation of $A$.

(b) $A$ recomputes $h(r)$ from the recovered value of $r$, and if this value is not equal to the value received from Token$BA$ then $A$ aborts the mechanism. If the computed value of $h(r)$ is the same as the value recovered from Token$BA$ then $A$ sets $D = r$.

(4) $A$ sends Token$AB = D$ to $B$.

(5) On receipt of Token$AB$, $B$ compares $D$ with $r$. If $r \neq D$ then the mechanism has failed and $A$ is rejected. If $r = D$ then $B$ accepts $A$.

NOTE 1 — Other information may be sent with either of the exchanges of this mechanism.

NOTE 2 — It is important that $B$ chooses the random number $r$ by a process which guarantees that the probability of the same number $r$ being chosen twice within the lifetime of the asymmetric key pair of $A$ is vanishingly small. If the same number $r$ is used twice, then a third party who intercepts the response $D$ on the first occasion that it is sent, will be able to impersonate $A$ to $B$ by replaying $D$ as a response to $B$ after $B$ has sent the number $r$ the second time. However, re-use of a previously valid value of $r$ will only invalidate one particular instance of use of the mechanism.

NOTE 3 — It is recommended that the key pair used in this mechanism be used only for the purposes of authentication, and should not be used for any other application (e.g. encryption of messages). If this recommendation is not followed then special care should be taken to prevent the verifier using the claimant as a 'decrypting oracle'; this could, for example, be achieved by requiring the hash-value $h(r)$ to be of a special form.

# Annex A
(informative)
Principles of zero knowledge mechanisms

## A.1 Introduction

In the context of the use of asymmetric cryptographic techniques, a potential weakness of an authentication exchange is that the verifier may abuse the mechanism to compromise the claimant's private key. When asymmetric cryptography is being used, the claimant uses the private key of his asymmetric key pair to compute the response to a verifier's challenge. The verifier may then, by choosing the challenge wisely, gain information about the private key of the claimant that could not have been obtained just from knowledge of the public key of the claimant.

This type of abuse of an exchange of cryptographic messages is known as using the claimant as an 'oracle', in that the claimant provides information about his private key at the behest of the verifier. The idea behind a zero-knowledge authentication mechanism is simply to remove this particular potential threat by careful design of the messages. This is done by ensuring that the verifier cannot use the claimant as an oracle.

## A.2 The need for zero-knowledge mechanisms

In applications involving modern computer networks, the need for security services such as user authentication, non-repudiation, etc. is widely recognized and steadily growing. In order to be able to use such services, it is necessary for a user to have access to private information, specific to that user. Examples are passwords, private keys to a digital signature system, etc.

It is of course mandatory for the security of the system that the private information stays private, i.e. does not leak to other potentially hostile parties. On the other hand, the private information must be used as input to the soft- or hardware modules that compute and send messages on behalf of the user. If the information is not properly used, the secrecy of the private information may be damaged, or even destroyed completely. An obvious example is when users identify themselves to a host by sending a password in cleartext. This reveals totally the private information with the immediate result that anyone eavesdropping on the line can impersonate all users whose passwords have been intercepted.

This is an example where too much information is being communicated. To illustrate this, note that from the point of view of the host, there are only two possibilities: either the user possesses the correct password or he does not. In information theoretic terms, this means that only 1 bit of information really needs to be communicated. By sending the entire password, we therefore communicate much more than is needed, and this is the theoretical background for the practical problem of eavesdropping.

It is natural to ask: can one design protocols for use of private information which communicate exactly the information they are meant to communicate, and nothing more? Informally, this is precisely the property that a zero-knowledge mechanism has. Consider for example a situation where user $A$ is assigned a key pair for a asymmetric cryptographic system $(P_A, S_A)$, such that $P_A$ is public while $S_A$ is private to $A$. Then using a zero-knowledge mechanism, $A$ can convince $B$ that $A$ possesses the private key corresponding to $P_A$, without revealing anything other than this fact. Since $A$ is characterized as the only user with access to $S_A$, this protocol can be used for authentication. In this case, the zero-knowledge property guarantees that $B$ will learn nothing that could help him to later falsely impersonate $A$.

The zero-knowledge property is achieved by designing a dialogue which can be simulated by the verifier alone. This intuitively proves that the verifier will learn nothing from the claimant in terms of properties of the claimant's private key, which the verifier could not have obtained himself from the corresponding public key. It also means that an observer to the exchange of messages making up the mechanism will be unable to decide if the claimant really was involved, or the exchange was simulated by the verifier.

Zero-knowledge mechanisms by nature require the use of asymmetric cryptographic techniques. Given the strict definition of a zero-knowledge mechanism, it is actually not possible to implement one. In fact, a much better description of the mechanisms in this part of ISO/IEC 9798 would be: 'Secrecy-protecting mechanisms'. However, the concept of zero-knowledge mechanism is part of a well-known and established theory in crypto-graphy, for which reason the terminology is used here.

## A.3 The definition

Going a little closer to a formal definition, a zero-knowledge mechanism takes place between two parties, a claimant $A$ and a verifier $B$. The claimant tries to convince the verifier that a certain statement is true. For example, this statement could be "I know the private key corresponding to $P_A$". To convince $B$, the claimant and verifier exchange messages for a while, after which $B$ decides to accept or reject the claimant's proof.

Three essential properties are needed for such a mechanism:

— **Completeness**. If $A$'s statement is true, then $B$ should accept it with overwhelming probability.

— **Soundness**. If $A$'s statement is false, then no matter how $A$ behaves, $B$ should reject it with overwhelming probability.

— **Zero-knowledge**. No matter how $B$ behaves, he receives only the information that $A$'s statement is true. A little more precisely: whatever $B$ receives when talking to a truthful claimant, $B$ could just as easily compute himself without talking to $A$ at all. What this means is that $B$ can simulate the conversation by himself, producing a conversation that looks exactly as if it had been generated by talking to $A$.

## A.4 An example

Consider the following example, which is a simplified version of the Fiat-Shamir mechanism, [3]. Here, we are given a modulus $n$ and a number modulo $n$, called $y$. In this case, $A$'s statement is "I know a square root modulo $n$ of $y$". Note that $x$ is a square root modulo $n$ of $y$, if and only if $x^2 \bmod n = y$.

The conversation between $A$ and $B$ goes as follows:

```
repeat $t$ times:

1. $A$ chooses $r$ at random (where
$2 \le r \le n-1$), squares it modulo $n$ and sends
the square to $B$.

2. $B$ chooses the bit $b$ equal to either 0 or
1 at random and sends it to $A$.

3. If $b$ equals zero then $A$ sends

                    $z = r$

to $B$. Otherwise $A$ sends

                    $z = rx \bmod n$,

where $x$ is the square root of $y$ modulo $n$,
which is known by $A$.

4. $B$ first checks that $z \neq 0$; if $z = 0$ then
$B$ rejects $A$ and aborts the procedure. If $b$
equals zero $B$ then checks that

                    $z^2 \equiv r^2 \pmod{n}$.

Correspondingly, if $b$ is equal to one then $B$
checks that

                    $z^2 \equiv r^2 y \pmod{n}$.

If the check is correct then continue, else
$B$ rejects and aborts the procedure.
```

It is not too difficult to see that if both $A$ and $B$ follow this procedure, then $B$ will never reject $A$; squaring $z$ means squaring either $r$ or $rx$, which will give the result $r^2 \bmod n$ or $(rx)^2 = r^2x^2 = r^2y \bmod n$.

On the other hand, if in any of the $t$ iterations, $A$ is able to give a correct answer to both $b = 0$ and $b = 1$, this means that $A$ can provide both $z_0$ and $z_1$, such that

$$z_0^2 = r^2 \bmod n$$

and

$$z_1^2 = r^2 y \bmod n.$$

By inserting the first equation in the second, it is straightforward to see that the number $z_1/z_0 \bmod n$ is a square root of $y$ ($z_0 \neq 0$ and $z_0$ must be relatively prime to $n$ with overwhelming probability). If $A$ can compute $z_1$ and $z_0$ with this property, then he can compute $z_1/z_0$, and so his statement that he knows a square root of $y$ is true. But conversely, if $A$ is cheating and does not know a root of $y$, he must be unable to answer at least one value of $b$ correctly in each of the $t$ iterations. Therefore the probability that a cheating claimant convinces the verifier is at most $2^{-t}$. For example, by doing 20 iterations, we reduce this chance to about 1 in a million. Thus the soundness property is also satisfied. As for zero-knowledge, note that all the verifier is left with after the conversation is over is two numbers $z$ and $r^2$, such that either

$$z^2 \equiv r^2 \pmod{n}$$

or

$$z^2 \equiv r^2 y \pmod{n}.$$

But this is indeed something that the verifier could make himself without talking to $A$. To do this $B$ just chooses a random $z$ and either defines

$$r^2 = z^2 \bmod n$$

or defines

$$r^2 = z^2/y \bmod n.$$

The fact that $r^2$ and $z$ are, in this case, computed in a way different from the way the claimant would compute them is insignificant; they are distributed in exactly the same way, i.e. it is impossible to tell the difference. Therefore, $B$ learns nothing he could not compute himself, except for the fact that $A$ knows a root of $y$.

Let us anticipate here an often asked question. If the verifier can make good looking conversations himself, without knowing a root of $y$, why should he be convinced when the claimant generates a similar conversation? The answer is that when $B$ simulates the protocol, he is free to generate the number in a 'backwards direction', i.e. to first choose $z$ and then find an $r^2$ that fits. In a real protocol execution, $A$ does not have this opportunity. The verifier expects to see $r^2$ before $b$ is chosen, and then the claimant must find a correct $z$.

Although we have glossed over a couple of technical difficulties here, these are the essentials of the argument why a mechanism has the zero-knowledge property.

## A.5 Basic design principles

The example from the previous section covers one of two basic design ideas that underlie almost all known zero-knowledge mechanisms, namely:

> The claimant sends a 'witness' to the verifier. Then $B$ asks $A$ one out of some set of questions. If $A$ is cheating, he cannot answer all possible questions, so we have some chance of catching him. On the other hand $A$ never answers more than one question, and this one answer alone reveals nothing to the verifier.

This design idea forms the basis of the mechanisms specified in clauses 5 and 6.

The other design idea, and one which forms the basis of the mechanism specified in clause 7, is based on the following:

> The verifier asks the claimant a question, for which the verifier already knows the answer. The protocol must ensure that this really is the case. If $A$ is honest, he can easily compute the right answer, but if he is cheating, he can do no better than guess at random, and will be incorrect most of the time.
>
> On the other hand, when $B$ receives the answer, he already knows what $A$ will say, and therefore the mechanism has the zero-knowledge property.

One easy example of this is when $A$ must prove possession of a private key in a public key system. The verifier can encipher a random message under $A$'s public key, and ask $A$ to return the deciphered message. Only the user knowing the correct private key can do this. To get the zero-knowledge property, we must ensure that $B$ really knows the message in advance. This standard contains an example of one way to do this, namely $B$ can be asked to reveal some information (the *witness*) related to the message.

A formal basis for a rigorous understanding of zero-knowledge protocols is given in [2] and [4].

# Annex B
(informative)
# Guidance on parameter choice

This annex provides guidance on parameter choice for two of the mechanisms defined in this part of ISO/IEC 9798.

## B.1 Parameter choice for the identity-based mechanism

Guidance is provided on the choice of the parameters $m$, $n$, $p$, $q$, $v$ and $t$. Note 2 in clause 5.2 of this part of ISO/IEC 9798 is also relevant to the choice of these parameters.

a) The modulus $n$. As stated in clause 5.2, the prime numbers $p$ and $q$ shall be chosen in such a way that knowledge of their product $n$ shall not feasibly enable any entity to deduce them, where feasibility is defined by the context of use of the authentication mechanism.

b) The public accreditation verification exponent $v$. Certain small prime values of $v$, e.g. 2, 3 or $2^{16}+1$ (in the latter case typically combined with $t = 1$ or 2) have some practical advantages in reducing the computational complexity of calculating the witness $W$; in general the choice of a relatively small value for $v$ will reduce the complexity of calculations for the claimant.

c) The multiplicity parameters $m$, $t$. The value $v^{-mt}$ is equal to the probability that a false claimant can succeed in a masquerade attack by 'guessing' the value(s) of the challenges $d_1, d_2, \ldots, d_m$ in advance in each of the iterations of the protocol. Thus $m$ and $t$ should be chosen so that $v^{-mt}$ is less than a probability threshold value which will depend on the sensitivity of the application. For most applications a value between $2^{-16}$ and $2^{-40}$ will be appropriate, where the exact value chosen will depend on the risk assessment.

The form of the identification data to be used in conjunction with this mechanism needs to be chosen with care, especially if identification data may need to be revoked or expired should the private accreditation information corresponding to the identification data become compromised. More specifically, expiry of an entity's identification data can be enforced by the inclusion of an expiry date in the identification data. This expiry information can be made available to the verifier by including it in the first exchange of the first of the $t$ iterations of the authentication procedure.

In a similar way revocation of an entity's identification data can be enforced by the inclusion of a serial number in the identification data. The verifier can then check this serial number against a 'blacklist' of revoked identification data.

## B.2 Parameter choice for the certificate-based mechanism using discrete logarithms

Guidance is provided on the choice of the parameters $d$, $g$, $p$ and $q$.

a) The prime numbers $p$, $q$ and the base $g$. As stated in clause 6.1, the values $p$ and $g$ shall be chosen so that, given an arbitrary integer $i$ ($1 < i < q$), finding an integer $j$ (if one exists) such that $g^j \bmod p = i$ shall be computationally infeasible. Such an integer $j$ is commonly known as the discrete logarithm of $i$ to the base $g$ modulo $p$.

Computational feasibility is defined by the context of use of the authentication mechanism. The bit lengths of the primes $p$ and $q$ provide lower bounds on the complexity of computing discrete logarithms, and thus the lengths of $p$ and $q$ must be chosen with care.

The prime $p$ can be chosen so that a copy of the binary representation of $q$ is embedded within the binary representation of $p$. Such an approach for choosing $p$ and $q$ may be useful in situations where storage space and/or communications bandwidth is at a premium. Annex C.2.1.1 provides an example of such a pair $p$, $q$.

If there is an odd factor smaller than $q$ dividing $p-1$, then the user key may be compromised by an attack of the type described in [7]. To prevent such an attack, $p$ and $q$ should be chosen so that $(p-1)/2q$ has no prime factors smaller than $q$. Ideally, $(p-1)/2q$ should be prime.

b) The challenge $d$. Suppose $d$ is chosen at random from the range $0 \leq d \leq 2^D - 1$ for some positive integer $D$. The value $2^{-D}$ is equal to the probability that a false claimant can succeed in a masquerade attack by 'guessing' the value of the challenge d. Thus $D$ (the bit length of $d$) should be chosen so that $2^{-D}$ is less than a probability threshold value which will depend on the sensitivity of the application. For most applications a value between $2^{-16}$ and $2^{-40}$ will be appropriate, i.e. a value for $D$ between 16 and 40, where the exact value chosen will depend on the risk assessment.

# Annex C
(informative)
**Examples**

This annex gives examples for the computation of the entity authentication mechanisms specified in this part of ISO/IEC 9798. Throughout this annex integers are given in hexadecimal representation.

The examples given here are intended for illustration and as an aid to validating practical implementations only, and should not be used in practice.

## C.1 Mechanism based on identities

### C.1.1 Example with public exponent 2

#### C.1.1.1 Parameter selection

In this example the public verification exponent $v$ is 2, which is even. Therefore the secret prime factors $p$ and $q$ must satisfy:

$$\gcd(\frac{p-1}{2}, v) = \gcd(\frac{q-1}{2}, v) = 1,$$

and

$$p - q \text{ is not a multiple of 8.}$$

$p$ = f859 cdc6 f78f d206 a8d2 e78c bfc8 2735 5798 5d16 cbf9 431f abfc c16f 9ca9 3a5e f099 d3e8 3fe0 c67e 31f5 77dd ccf1 8287.

$q$ = fef3 6abf 2aaf afa7 1c0b ca24 efe2 fb28 3366 1fb9 266f 9046 3c78 aa54 4a7c e2d8 9e56 071e 42db 00b3 c87e dc89 563a 02fb.

The public modulus $n$ is 768 bits long.

$n$ = f755 3ef8 611b c569 0a2e 4d13 801a 94be 4dc8 fe2d da6c 6e11 586e 1941 81fb 96bf dc09 4d04 edbe ed1d 22ce 1fae 689b a233 3298 7fd7 9ef8 715f 1f5a 5eb4 b41f 45ea fcc5 4f32 5f21 5135 2930 8ff9 d8cd 5738 3801 fce9 7b51 f50f 8192 b0e1 c066 085d.

$k_s = 767$.

The accreditation authority's private accreditation exponent $u$ is the least positive integer satisfying: $uv + 1$ is a multiple of $\frac{\text{lcm}(p-1,q-1)}{2}$.

$u$ = 1eea a7df 0c23 78ad 2145 c9a2 7003 5297 c9b9 1fc5 bb4d 8dc2 2b0d c328 303f 72d7 fb81 29a0 9db7 dda3 a459 c3f5 cd13 7446 2769 68ea 2f97 1df6 2b4f 75a0 608e 8471 ae38 da4c 4d97 0fb9 e817 6486 be34 e740 1522 443c 5f12 c5bb b0e3 cb8f 53a7 505b.

$m = 8$. $t = 3$.

#### C.1.1.2 Identity selection

The identification data consists of a sequence of $m = 8$ parts. These identity parts are constructed using the string 'Alex Ample' postfixed with a 16-bit field number.

$I_{A1}$ = 416c 6578 2041 6d70 6c65 0001
$I_{A2}$ = 416c 6578 2041 6d70 6c65 0002
$I_{A3}$ = 416c 6578 2041 6d70 6c65 0003
$I_{A4}$ = 416c 6578 2041 6d70 6c65 0004
$I_{A5}$ = 416c 6578 2041 6d70 6c65 0005
$I_{A6}$ = 416c 6578 2041 6d70 6c65 0006
$I_{A7}$ = 416c 6578 2041 6d70 6c65 0007
$I_{A8}$ = 416c 6578 2041 6d70 6c65 0008

#### C.1.1.3 Accreditation generation

$J_{A1}$ = 5341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e301 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e301 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e301 9141 276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00 e316.

$C_{A1}$ = 79b7 7f76 b264 a2e0 bc4c e8f9 f29a 2175 99b4 2567 6dda 9360 228b ede5 748a d735 b2e9 bcf8 de99 6c8a 87db f920 26f4 b81e f97f 2b18 e50c 526b 2a40 f619 7d72 d7da 7d2e a641 2c2a fd97 df62 dfc4 56eb b043 8a99 1880 8749 387a 4a52 4a78 5049 4be6.

$J_{A2}$ = 29a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7281 49a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7281 49a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7281 48a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7293.

$C_{A2}$ = 41fb 8c2e c141 60fc 896b 1f36 d68a 4f8e 7a31 1226 31e2 28ea 568e c98e b09a 0e88 3500 21c1 8ac6 f81a 9f29 e8d2 25b0 8795 40b8 1791 e0ff 0cab 4aca 6e7c e17d c59c bc7e c931 9d92 beb5 8433 111e 14fd f601 6494 536f 2bc9 c692 a1f0 1da5 bd5d 8b90.

$J_{A3}$ = 29a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7401 c9a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7401 c9a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7401 c8a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6 1232 f700 741b.

$C_{A3}$ = 03c1 c485 28ac 5b9b 639a 0123 c093 6f2e d642 89e2 799a a434 2377 92d3 3ef2 d055 2cd8 3cfc 68ea 6a70 e310 1e72 5724 9c92 48e9 fd19 7ff1 d126 4c62 2ba8 8cac f99f daed 0adc e206 7e29 9dd9 0a15 3868 5d3b 396a af56 0332 fc84 46d0 2ab5 69fa 6cc6.

$J_{A4}$ = 5341 276c 2465 f078 5e20 9341 2a6d fe70 276c
2465 ee00 e904 9341 276c 2465 f078 5e20 9341 2a6d
fe70 276c 2465 ee00 e904 9341 276c 2465 f078 5e20
9341 2a6d fe70 276c 2465 ee00 e904 9141 276c 2465
f078 5e20 9341 2a6d fe70 276c 2465 ee00 e946.

$C_{A4}$ = 0e78 f7fe d61e 0934 ce1d 5c30 e3d8 7e50 def0
f339 06ff cadb ac33 fced 95c4 8579 d651 33fe 1bd6
963a 8a1e 56c5 d318 a94c fdab 5c27 9a61 25c5 07ed
d1da 4aec a673 47cc 78f6 6a2b a671 e432 5d94 3b18
ad1e a2f2 f51e 2301 5070 ede3 fd92 5291 1ea2.

$J_{A5}$ = 29a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6
1232 f700 7202 c9a0 93b6 1232 f83c 2f10 49a0 9536
ff38 13b6 1232 f700 7202 c9a0 93b6 1232 f83c 2f10
49a0 9536 ff38 13b6 1232 f700 7202 c8a0 93b6 1232
f83c 2f10 49a0 9536 ff38 13b6 1232 f700 722b.

$C_{A5}$ = 2be9 2edd 3b6c 4977 ffe7 3b6f d0c9 1835 1b04
2b0a 33b9 7fb1 d407 724c 5035 a335 109e 791f a4b7
03f2 d8be 9a8e 031e bad6 7175 90c7 ad03 9250 9f4b
177b 40f6 0653 9eb7 6d1d 49e8 949e bc12 989f ad28
675e 8dd2 eb59 e5fc 4703 2ef1 b9a1 da84 b8d1.

$J_{A6}$ = 5341 276c 2465 f078 5e20 9341 2a6d fe70 276c
2465 ee00 e206 9341 276c 2465 f078 5e20 9341 2a6d
fe70 276c 2465 ee00 e206 9341 276c 2465 f078 5e20
9341 2a6d fe70 276c 2465 ee00 e206 9141 276c 2465
f078 5e20 9341 2a6d fe70 276c 2465 ee00 e266.

$C_{A6}$ = 1ae7 264d 6b92 9c8d 3131 5411 e0b0 65c1 9ac2
c815 a6dc 92bd 26e8 2281 8ce8 d9b0 bde2 b895 a267
f6eb 226e 3989 fed6 fac0 1865 66f0 9bed 5992 b882
02c1 2df7 e903 3849 4881 8570 298d 8df1 27c4 4758
f769 ccf4 a6ce 2303 3fb0 b13c 17c9 8eb7 7db4.

$J_{A7}$ = 5341 276c 2465 f078 5e20 9341 2a6d fe70 276c
2465 ee00 ef07 9341 276c 2465 f078 5e20 9341 2a6d
fe70 276c 2465 ee00 ef07 9341 276c 2465 f078 5e20
9341 2a6d fe70 276c 2465 ee00 ef07 9141 276c 2465
f078 5e20 9341 2a6d fe70 276c 2465 ee00 ef76.

$C_{A7}$ = 496f db6e e535 0180 1c86 6610 8769 f225 4631
6b07 0267 d57d 7613 f42c 70e1 0d39 256f 5c60 3d22
7f28 5401 0d42 05d3 eab1 f795 b386 5595 4234 7f8a
3ce0 b483 8d7f 0cf2 dea6 f895 277a a2f5 1732 e854
8f44 d31a e502 ad03 7194 0d65 ab07 c55d c85f.

$J_{A8}$ = 29a0 93b6 1232 f83c 2f10 49a0 9536 ff38 13b6
1232 f700 7004 49a0 93b6 1232 f83c 2f10 49a0 9536
ff38 13b6 1232 f700 7004 49a0 93b6 1232 f83c 2f10
49a0 9536 ff38 13b6 1232 f700 7004 48a0 93b6 1232
f83c 2f10 49a0 9536 ff38 13b6 1232 f700 7043.

$C_{A8}$ = 4892 3caa be40 0643 ce8a 24ab 1e48 5876 ae94
9ae1 a465 ced2 a59e 63d2 fd37 044b 202e 1543 64db
38b5 d16a b675 2401 98be 23c1 ba7c ca8a 0058 4ae9
e637 f70a a640 c1a8 a1b3 2f5a 3a35 7e75 7df4 04e3
0ffb 8203 0b5c 986a 131b 1aa7 b37b 2c04 4dbe.

### C.1.1.4 Authentication exchange

The authentication procedure is iterated $t = 3$ times.

#### Iteration 1:

*Step (1):*

$r$ = f637 29e2 8723 3b12 d7c9 b048 8626 7680 3880
f8b8 0ba0 3497 60fe 2c2d ee4f a8ed 7860 8a3f 24f1
22f4 45d1 cb18 8ef2 82c1 a6ea 4453 c550 cfcf d16f
ebdb add9 51ed 750e d717 cd83 8cd3 1cbc ce82 c2e6
afe4 8507 e66b 5417 33eb d4e1 8c94 e180 e1c6.

$W$ = 573c 7004 78cd feac 4025 4777 20fa 7e68 8010
deb9 0a10 676e e59b 9074 40be e961 86af f988 6449
1a34 aa7e b741 1af9 3e12 fd4b d9ff fc0b dfc9 5c1a
3780 8b77 aa8f 3170 e240 c6ae 9d03 c7d9 9acc e21b
f125 a4f7 9f16 f26d e6bf dcb2 d6ae 6187 d536.

*Step (3):*

$d_1, d_2, ..., d_8 =$ 0, 0, 1, 0, 1, 1, 0, 1.

*Step (5):*

$D$ = 415c d169 1334 412a be2d 6bdd d373 305f 0eed
f1d2 f943 68e6 62cb da0e 9e46 8840 af85 9372 8fb5
6354 3bb5 7061 ff42 210d 6fbf 8232 3ef6 82a3 1956
2d57 3e06 0a6c 5c63 9762 a18a f0ca 12de e2bf f91d
c376 844a 7f14 6113 c6a3 269c 8211 2d62 cbc5.

*Step (7c):*

$W'$ = 573c 7004 78cd feac 4025 4777 20fa 7e68 8010
deb9 0a10 676e e59b 9074 40be e961 86af f988 6449
1a34 aa7e b741 1af9 3e12 fd4b d9ff fc0b dfc9 5c1a
3780 8b77 aa8f 3170 e240 c6ae 9d03 c7d9 9acc e21b
f125 a4f7 9f16 f26d e6bf dcb2 d6ae 6187 d536.

#### Iteration 2:

*Step (1):*

$r$ = 6d5f 2b2f 5027 39a9 177d 30f4 8d5e 1b6d 40d4
eea9 35d8 3bb4 0288 b447 6cdf 3f5c f0f9 b714 08dc
eaae ff5a b380 c96b acb0 973b 78ab 08f6 d085 795d
92e3 630b ffa8 59ae 1eb5 2b52 0c5d 6030 fe80 a4e1
1a2f ed3d 6801 5311 a0ab 5018 8a73 f1b6 9460.

$W$ = 44ce 41f4 9c42 ec82 f5ac 5a42 03e6 6cc5 bcb6
f343 f0db 0c07 5318 fc26 328f 1f07 35df 653b c8bc
9bf9 a6af fafe 98a5 14d6 4952 1f5b c1a9 de55 721e
8950 399d 7c3b cb2d ba2d ef50 5cce a17e e8ec 314a
6621 ccd4 5755 5136 b4f6 fdb6 b6de ce3d 94c9.

*Step (3):*

$d_1, d_2, ..., d_8 =$ 0, 1, 1, 0, 0, 0, 1, 0.

*Step (5):*

$D$ = 0c84 7739 615a 2c11 a240 6314 bb0c 1bce 155f
b99e 7c92 8c3d 15a8 ee81 ea67 1f6d c248 d1c8 9c06
a80f b4e2 a2cb d5e3 54ca b5ba 98f6 f2ac e1f6 c160
dc82 8f8e db4f 2131 a11c 86a0 86ba 9f74 e838 b653
cd25 02cc 1877 1ed7 113c 1cb3 bf19 6cac 738f.

*Step (7c):*

$W'$ = 44ce 41f4 9c42 ec82 f5ac 5a42 03e6 6cc5 bcb6
f343 f0db 0c07 5318 fc26 328f 1f07 35df 653b c8bc
9bf9 a6af fafe 98a5 14d6 4952 1f5b c1a9 de55 721e
8950 399d 7c3b cb2d ba2d ef50 5cce a17e e8ec 314a
6621 ccd4 5755 5136 b4f6 fdb6 b6de ce3d 94c9.

### Iteration 3:

*Step (1):*

$r$ = c50a 4b30 b2ad 7b7a 26ac 6ca5 0b2e 2d2c 0d40
1fb8 4e6d 6d12 3fce c8f2 9f55 26cf eced cbf0 184c
f826 5db5 db87 a82e 9397 9fd5 9152 b65a fdbd f5a2
c017 9781 33ab 12ec f85f 5db8 9fdb 6aa5 43b5 87b2
88f0 2963 4604 9703 5838 7cb3 28bd a1a9 b699.

$W$ = 0268 2fb9 c79b 2c9f bdc3 2804 6dc5 9a30 d3c7
0e02 01db e43e 2c09 8fde f967 037f 20be 354e c92d
208e ecfd a688 7126 58ef 28fd e27c c97b 8520 a408
1570 0539 de84 632b 0ba2 b899 95c9 199e 9d61 a0cb
c036 2ed1 0a8e d566 4935 98cb 7f32 038d 525d.

*Step (3):*

$d_1, d_2, \ldots, d_8$ = 1, 1, 0, 0, 0, 0, 0, 1.

*Step (5):*

$D$ = 3c61 982e fa0e 5c75 dada 504e d5e2 b056 e3cf
80bb fad1 2925 05bd 426c 952e bace ffd3 cdb3 cb5d
2233 7128 9507 81a0 464a f7cf db3e dbaa f76f 2d1a
d3c8 7ce2 5289 fe4d 87eb 9583 20ba 749a 369a da50
0227 bd8a 8439 8e8c 5a4e 82eb 90a1 0294 2448.

*Step (7c):*

$W'$ = 0268 2fb9 c79b 2c9f bdc3 2804 6dc5 9a30 d3c7
0e02 01db e43e 2c09 8fde f967 037f 20be 354e c92d
208e ecfd a688 7126 58ef 28fd e27c c97b 8520 a408
1570 0539 de84 632b 0ba2 b899 95c9 199e 9d61 a0cb
c036 2ed1 0a8e d566 4935 98cb 7f32 038d 525d.

## C.1.2 Example with public exponent 3

### C.1.2.1 Parameter selection

In this example the public verification exponent $v$ is 3, which is odd. Therefore the secret prime factors $p$ and $q$ must satisfy:

$$\gcd(p-1, v) = \gcd(q-1, v) = 1.$$

$p$ = a0ca e977 6bc5 5a7f f591 7bc8 8164 16f9 503c
16e9 0a0c 4da3 b1d3 d97a 1220 605e 071f 1c6f 9305
def5 4832 0ea3 5e76 4d45 698e 9196 09a4 35f1 fde4
0d7c 3146 8eb3.

$q$ = e349 3f5b 7808 aac9 6083 b0b6 d97d 5a57 d300
43c8 6416 719e 2d95 7654 5f0a c7b1 4061 8232 728c
7777 0fbe aac2 f5f0 8238 5783 91bb ceb5 be1e cd31
b043 be4f 75df.

The public modulus $n$ is 1024 bits long.

$n$ = 8ec1 eeac da97 aa8b a6e2 fb76 4423 dcc7 d723
2848 5219 c685 bcef f9a8 f970 a9b2 ed4d 7dd8 64dd
162d f77a a9b8 549e 7029 d409 9494 61af 3590 e5ca
6cf4 1f5b 073d b399 f566 0068 4ff9 60f8 8336 85a6
d337 84c3 cade c2e9 32fe a1fe 9b05 85b2 8a8a 4b02
4bdb 7d46 9b62 2657 b19a ade2 768d 3608 f0be 09bb
9d59 c88c 7d3c 0eeb 1ced.

$k_s = 1023$.

The accreditation authority's private accreditation exponent $u$ is the least positive integer satisfying: $uv + 1$ is a multiple of $\mathrm{lcm}(p-1, q-1)$.

$u$ = 2f95 fa39 9e32 8e2e 8cf6 53d2 16b6 9eed 47b6
62c2 c608 9781 e9a5 5338 5325 8de6 4f19 d49d 76f4
5cb9 fd28 e33d 718a 2563 46ad dc31 75e5 11da f743
79a6 b51e 57be ba81 eedb b433 6e3a ae4b c792 6397
20a2 2082 7ab9 c6ec d13e eb87 1912 5c2d 20d3 abd5
e468 7d3c 16fc 9a22 52bc 1dd3 e25a 7c52 4479 65cb
386d a9d2 3fd4 0a71 b2c9.

$m = 5$. $t = 5$.

### C.1.2.2 Identity selection

The identification data consists of a sequence of $m = 5$ parts. These identity parts are constructed using the string 'Alex Ample' postfixed with a 16-bit field number.

$I_{A1}$ = 416c 6578 2041 6d70 6c65 0001
$I_{A2}$ = 416c 6578 2041 6d70 6c65 0002
$I_{A3}$ = 416c 6578 2041 6d70 6c65 0003
$I_{A4}$ = 416c 6578 2041 6d70 6c65 0004
$I_{A5}$ = 416c 6578 2041 6d70 6c65 0005

### C.1.2.3 Accreditation generation

$J_{A1}$ = 676c 2465 ee00 e301 9341 276c 2465 f078 5e20
9341 2a6d fe70 276c 2465 ee00 e301 9341 276c 2465
f078 5e20 9341 2a6d fe70 276c 2465 ee00 e301 9341
276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00
e301 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c
2465 ee00 e301 9141 276c 2465 f078 5e20 9341 2a6d
fe70 276c 2465 ee00 e316.

$C_{A1}$ = 2f98 686f a57f 0799 f0a6 42dc 20ae 91f4 f875
a346 a6b8 e951 042e 77c6 1ad9 0a60 915d 8ea6 9dbf
bec2 589f 331f 26a7 0859 9ca8 27b5 5a5c b78b ee4e
07b4 9c86 dd7e afea 5a4e 9b9d 068b 173d a27e ebc0
78b1 8305 e930 48db b81e 49cd 83f0 e101 260a 76bc
10d8 8679 5b4c b096 7943 5195 ea43 b98b 35a4 c3b1
eb02 8e7c 0f54 0949 67f8.

$J_{A2}$ = 676c 2465 ee00 e502 9341 276c 2465 f078 5e20
9341 2a6d fe70 276c 2465 ee00 e502 9341 276c 2465

f078 5e20 9341 2a6d fe70 276c 2465 ee00 e502 9341
276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00
e502 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c
2465 ee00 e502 9141 276c 2465 f078 5e20 9341 2a6d
fe70 276c 2465 ee00 e526.

$C_{A2}$ = 31ca f31b 374b a31c ba12 b38a 23e8 46c7 dac1
7ead 4647 34e9 8adb 781b 4c56 2463 ced6 9ee3 6eae
0992 c2af 5027 5874 7b15 4f2d 723d a590 593a ec39
5fbd de29 7bc5 0fab 1af9 7143 a0d1 b25a b1bc 6c7f
544f 72a9 35ec 265d a54d d52d c5b6 cde9 58ef 593b
2dfa 5ba6 0c05 8f11 6c9f 488e d9fb 8680 112c ba35
aec8 7441 7aa5 8f4b 57f5.

$J_{A3}$ = 676c 2465 ee00 e803 9341 276c 2465 f078 5e20
9341 2a6d fe70 276c 2465 ee00 e803 9341 276c 2465
f078 5e20 9341 2a6d fe70 276c 2465 ee00 e803 9341
276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00
e803 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c
2465 ee00 e803 9141 276c 2465 f078 5e20 9341 2a6d
fe70 276c 2465 ee00 e836.

$C_{A3}$ = 2dac 4f99 c5c7 6a01 56bf 01f1 d135 2b07 3742
cc23 9e43 eaed d6d3 0e9f 8c17 750d 0024 5099 8caa
7cf6 526c adc6 cd78 74d9 90b3 bcbb abc0 603f 0431
ae82 81f0 2d92 4c8b 3add 7eb0 2c37 bca5 2ea5 c740
6752 0bdc 0644 a64e ae8e 6690 4ac0 31af cc8d 1c8b
1a4f 8c04 de6a bb29 3d98 7449 0a87 56e6 c54d 0259
07a3 136c a560 3c42 d0a1.

$J_{A4}$ = 676c 2465 ee00 e904 9341 276c 2465 f078 5e20
9341 2a6d fe70 276c 2465 ee00 e904 9341 276c 2465
f078 5e20 9341 2a6d fe70 276c 2465 ee00 e904 9341
276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00
e904 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c
2465 ee00 e904 9141 276c 2465 f078 5e20 9341 2a6d
fe70 276c 2465 ee00 e946.

$C_{A4}$ = 10d9 1f22 933c 27c2 589b b1d0 0a27 767c 1465
b9de 1074 aaf5 e845 128c d422 8cb1 2688 7139 2526
7d7b 6e07 cf01 975e aeac 9b50 9e8b 2432 06de b57c
0676 5069 5ddb 2a53 419d 7703 30ca f37a e3d7 148e
8dab b8c9 dd68 359a cb64 7d4c bbf1 156b 5aa8 49f0
7e12 3e0f d71c 853a d005 a36b 9bf9 ee65 3e12 cc18
4fa1 a3b8 9ee2 a121 353c.

$J_{A5}$ = 676c 2465 ee00 e405 9341 276c 2465 f078 5e20
9341 2a6d fe70 276c 2465 ee00 e405 9341 276c 2465
f078 5e20 9341 2a6d fe70 276c 2465 ee00 e405 9341
276c 2465 f078 5e20 9341 2a6d fe70 276c 2465 ee00
e405 9341 276c 2465 f078 5e20 9341 2a6d fe70 276c
2465 ee00 e405 9141 276c 2465 f078 5e20 9341 2a6d
fe70 276c 2465 ee00 e456.

$C_{A5}$ = 171d fcf1 dcda 63b1 a53c 1f8b a8a7 d46c 8ced
c942 956d 9fea 5469 61dc cc52 dcd8 4821 604e bde1
cd25 d23d 7815 0bdc 85b3 7aa0 b691 9609 6aae 6ca3
e843 5575 4eb1 2470 234d 05ea 8a46 45c2 fa69 eaac
e374 78d2 7394 12db 57c7 9018 6a85 8a00 8e56 732d
71f3 f5ea dc8e 0988 95cf 3a50 cfd8 e106 b640 2c9c
9c07 c630 6e15 ead4 e050.

### C.1.2.4 Authentication exchange

#### Iteration 1:

*Step (1):*

$r$ = 0fdf 125c 140e 2a4f 54d4 ef3b 41a4 e11e 7175
4f7b f003 1d01 8a51 8ac2 27be 49fa 6987 e987 ebb1
0ebb dbfd ee1a 9443 7313 fc53 44e0 5238 0a7f 155a
9ad0 70fd e4ef eb37 e6fb 1a82 d078 bc73 31a1 b95b
58e2 3665 ce63 1300 e862 2d52 5552 dd45 fdde e10e
f0e5 5d63 106c 0692 5dd8 4b1c ba98 fa0c 4e57 1a0b
135e c5d2 9659 47bf 7145.

$W$ = 2935 3492 54b4 b32c fa87 b7af 0a17 12ef 568b
c5d9 593d f8a6 2972 b911 18c1 9d2e 5cee 57a5 1318
d878 e829 e9e7 0306 f482 e81f 57e4 e18d d71d 5312
a0a7 3539 971a e02f 46b4 1e8a 456e 9260 4090 8018
9c7a dfea d147 d75c 2aea 143e d666 c3b4 3993 616b
f040 c310 7baf 4584 c2e7 ec50 b8b2 040d c803 19c4
4faf 7b00 b3b0 67ab 8ee2.

*Step (3):*

$d_1, d_2, \ldots, d_5$ = 2, 1, 2, 1, 2.

*Step (5):*

$D$ = 2381 3185 d61b 1684 1c72 8b6d eb9e 30a6 6b7f
c509 3c00 c5de 6194 9ba2 9721 a8df fe0d 04e5 f5c5
2e71 ac97 b511 a144 d4e9 7d5b 359d 1e92 ffb9 ca3a
2e61 42a8 a0a2 b3cd 22ee 1e28 5432 d645 21d4 73f6
d9b0 5765 22fb 3ea0 78ba 1ee4 f70c 8d56 d354 df85
a7e3 1257 f9c2 2865 d0e9 2f8a e587 61b0 4f04 4dd2
8b74 cf60 b332 d461 e5c7.

*Step (7c):*

$W'$ = 2935 3492 54b4 b32c fa87 b7af 0a17 12ef 568b
c5d9 593d f8a6 2972 b911 18c1 9d2e 5cee 57a5 1318
d878 e829 e9e7 0306 f482 e81f 57e4 e18d d71d 5312
a0a7 3539 971a e02f 46b4 1e8a 456e 9260 4090 8018
9c7a dfea d147 d75c 2aea 143e d666 c3b4 3993 616b
f040 c310 7baf 4584 c2e7 ec50 b8b2 040d c803 19c4
4faf 7b00 b3b0 67ab 8ee2.

#### Iteration 2:

*Step (1):*

$r$ = 583e 235e d777 a918 1d4a a2f9 da89 a905 3e65
a827 d573 c68e 73e7 ce5a 61b1 d135 1d6c 4236 df99
2c1a d174 588b 8e07 5f14 c954 44cf 3493 b44e 3971
dadc 38e6 3a86 a1c8 e4dc 222e 5290 86a3 3301 cd87
5be2 3c5b d963 df15 3f4b add7 4ea5 3eb5 f7a5 41d3
90a0 99b3 78ad 46ac 635a 3bdf 72eb 959d 4e1c ca8c
12aa 4a34 38ce a016 556f.

$W$ = 2f69 40df a78e 740f bf64 9f12 f55d 5081 ab45
d794 b08b c3f9 c98e d1de f793 7fa5 9b20 860b 9474
bd0e 4874 93b4 ec52 6c31 7e2f c250 3a4a 4951 1b0a
9b12 8e53 80a8 b58d beca 35c2 d633 5b80 f184 89fb
eed9 b3da f95f 71b2 0eb1 f6e9 c4f6 2054 7eff 0ac5

97b5 7fc3 7d89 e746 91d1 8517 440a 1d37 63fd 377d 09c7 5369 88c5 9389 d3f3.

*Step (3):*

$d_1, d_2, \ldots, d_5 =$ 1, 1, 0, 0, 0.

*Step (5):*

$D$ = 10bf ace9 3bc3 80e1 f8a8 9229 f66f d7af 5819 d745 be0b 7980 8df0 0692 2e8f bdef 8650 d311 2269 eae1 7bc2 a641 509d 1238 148d d07c ca97 3d1a 67c5 e34c 50e2 9ab8 52eb 42e7 e527 6eb3 6cfb 18dc ec2f 6939 9461 7f05 6320 94b7 8c07 87cc db53 8f85 d8be 6754 e39b 9f2e 30cc 935f c2da 97cd 60c6 99ba bfac 07fe 02ea 406a 613e 34ea.

*Step (7c):*

$W'$ = 2f69 40df a78e 740f bf64 9f12 f55d 5081 ab45 d794 b08b c3f9 c98e d1de f793 7fa5 9b20 860b 9474 bd0e 4874 93b4 ec52 6c31 7e2f c250 3a4a 4951 1b0a 9b12 8e53 80a8 b58d beca 35c2 d633 5b80 f184 89fb eed9 b3da f95f 71b2 0eb1 f6e9 c4f6 2054 7eff 0ac5 97b5 7fc3 7d89 e746 91d1 8517 440a 1d37 63fd 377d 09c7 5369 88c5 9389 d3f3.

## Iteration 3:

*Step (1):*

$r$ = 170d 20a2 00f4 7124 fac0 695d 1612 07ce 848a cba5 d1f5 adcb 0ab1 ae12 6377 4b5a 4ea1 48e2 434a 804b e374 6787 a775 29c9 59de c8c5 2952 cade 4d81 3061 6f6d f1ba b7b6 2b24 be92 5cfc 44e6 e2df 2b97 31c2 180f 3b32 c068 ee9f 0b70 88e0 e700 67d9 d799 0622 b329 bea3 c1f3 ff80 e614 3562 794a a065 28e8 61dd fc41 109c 6080 ded0.

$W$ = 2c81 4f76 98b9 4169 b7c5 14a4 6f3c d9e9 5f7c 7aa1 9de9 9d59 fd42 7d8a 6f96 a45c 345c 823e e7ba 6dee 98a0 2c44 34b9 4249 32df 3ef8 4f79 4f8a 69d5 c970 a5f8 f8bc f1f7 4189 56a9 e9aa ae01 7928 2ccf c1c8 5c75 1a7f a117 6c66 366d 8ee2 ef5d 9145 0aa1 18e4 1bc1 6601 373a 1340 72ba adb6 1565 e292 d826 0047 afb8 0b20 f227 2290.

*Step (3):*

$d_1, d_2, \ldots, d_5 =$ 0, 0, 1, 0, 2.

*Step (5):*

$D$ = 10fc 250c 11ad 61b8 19b1 eaa3 9376 4e22 cfce 7b98 af17 0d1e b74b a5c7 596b dafc f953 7fa6 c6d7 6fe9 61c1 c7bf 087b eb8d 25c8 74d9 ce81 959e f419 1912 927f da1c 04f7 92a6 dd1a 0ca3 7f86 2dc6 c4b9 331f 69a7 cd46 c7ad 7352 43d4 6223 ba0a c578 aaf4 1443 a413 67e7 5497 47e3 7603 ecad e131 f18a 771c edd2 064c 5d56 11a9 7fa1.

*Step (7c):*

$W'$ = 2c81 4f76 98b9 4169 b7c5 14a4 6f3c d9e9 5f7c 7aa1 9de9 9d59 fd42 7d8a 6f96 a45c 345c 823e e7ba 6dee 98a0 2c44 34b9 4249 32df 3ef8 4f79 4f8a 69d5 c970 a5f8 f8bc f1f7 4189 56a9 e9aa ae01 7928 2ccf c1c8 5c75 1a7f a117 6c66 366d 8ee2 ef5d 9145 0aa1 18e4 1bc1 6601 373a 1340 72ba adb6 1565 e292 d826 0047 afb8 0b20 f227 2290.

## Iteration 4:

*Step (1):*

$r$ = 4224 ffa1 41bc 4bea 93d0 14ec acc1 fa5d 9616 f0fa c12a a029 3f84 a858 f916 13c3 ccef e0d1 16e5 30fc 4fda 72f5 15ce 5996 e211 fcc8 eee0 0719 84a6 b717 a7be cc05 afd1 8b8e 71ec 2d3f 285c 6d07 39c6 b4ef 3660 468d c13f 24c8 0ef8 c992 59f0 04c8 996e 9387 99a7 0769 03d7 fd23 9472 3396 6cc0 2ec3 fc30 33b9 4a8c d1fb 919f 610a.

$W$ = 0525 c5ba f6f9 78b1 1fd6 6e86 0de3 ebd3 314e efb0 9ca7 8193 3b2e cb5f 8046 1b46 a87a 727a a317 e163 9dc4 2b55 8202 65dd 0f2c ce4d 57fe 84dc 08f5 49ee 896a a897 c29a dabb e0c6 78fe 0be5 753d 0d97 0d99 f3d2 1eef b822 7715 39e7 402d 63da 03d4 b66f 3789 066e 4c36 d025 3466 1b6a e359 f290 bb21 c1fc 01fe 23fb 0135 77d8 0ea9.

*Step (3):*

$d_1, d_2, \ldots, d_5 =$ 0, 0, 0, 1, 2.

*Step (5):*

$D$ = 3559 cda5 4d03 1913 3dca c484 3a3f 9635 86e1 8455 3448 e759 1213 5269 9d8a 5a5d 4c4e 178e befd 7223 2808 356a 427a 0b1e 1681 3ba4 c564 27bf 0c03 1fb9 e35e ebcc 51c3 bfb2 346c 11f6 bc7c 3dfc d3f5 000d 2614 1ad5 7197 4362 f195 9750 a8ae 5750 f36d fd40 b930 85ca 4600 b753 0e36 9791 4c87 4860 56a9 ad71 4717 eebc 3dc3 633e.

*Step (7c):*

$W'$ = 0525 c5ba f6f9 78b1 1fd6 6e86 0de3 ebd3 314e efb0 9ca7 8193 3b2e cb5f 8046 1b46 a87a 727a a317 e163 9dc4 2b55 8202 65dd 0f2c ce4d 57fe 84dc 08f5 49ee 896a a897 c29a dabb e0c6 78fe 0be5 753d 0d97 0d99 f3d2 1eef b822 7715 39e7 402d 63da 03d4 b66f 3789 066e 4c36 d025 3466 1b6a e359 f290 bb21 c1fc 01fe 23fb 0135 77d8 0ea9.

## Iteration 5:

*Step (1):*

$r$ = 0d2c cc3f 814b a506 7753 4948 8ba7 c8de 8a06 8da8 2eba 8b9f fc7e ab4f f439 8317 91b0 2700 28c8 c170 e8ec 16ca f279 08ba de15 912f dbf4 0562 0b1d f95b b16e b24a 2b46 e0d7 e466 c636 684b e07d d57f c67b e66d 6b16 af88 d873 047a 94e1 be1c 639f 2426 efb5 614b fbeb 00c4 d6af 0d14 0ac5 d54f c632 f933 80e6 e94d 5a35 675f fa6d.

$W$ = 23d8 0778 c35c eccb b47f 8d83 881d 5e18 dfb8
a4e7 33af e9c8 af80 ef7b e6c0 a9b5 3d3c a4f3 524b
f72c 18b6 edb7 ba71 4523 9d48 4463 a817 9f18 83cf
71e9 96dc da61 58a2 3936 91bf 133b fdfe e906 b713
64b6 cdef 8446 2be2 8634 67d0 0f78 7a67 9ba4 b42f
92d1 ea7d 8c5e bfb9 8b7f d049 3907 61a8 1300 a9d1
4e18 cf4b dc08 064a ddcb.

*Step (3):*

$d_1, d_2, ..., d_5$ = 0, 1, 2, 1, 0.

*Step (5):*

$D$ = 1061 f560 095a f8d8 81ae 7844 aa56 62cd d651
4c40 4f1a 4cb1 ebc3 0672 7ccc 4c10 0ded 8447 e1b0
5611 7135 78f3 86fd 6251 59ef 5755 c26f 7780 9fa9
a601 d059 b755 4cf9 bfa8 f491 6ce4 96c1 c082 c9d9
8622 b035 4341 3171 fa8c 6be3 c07a 418c b828 02f3
05a2 6998 40dc a632 ef9c a5a8 c9b3 e81e 902d 6699
4c00 4efa 3ce2 766d 42c1.

*Step (7c):*

$W'$ = 23d8 0778 c35c eccb b47f 8d83 881d 5e18 dfb8
a4e7 33af e9c8 af80 ef7b e6c0 a9b5 3d3c a4f3 524b
f72c 18b6 edb7 ba71 4523 9d48 4463 a817 9f18 83cf
71e9 96dc da61 58a2 3936 91bf 133b fdfe e906 b713
64b6 cdef 8446 2be2 8634 67d0 0f78 7a67 9ba4 b42f
92d1 ea7d 8c5e bfb9 8b7f d049 3907 61a8 1300 a9d1
4e18 cf4b dc08 064a ddcb.

### C.1.3 Example with public exponent $2^{16}+1$

#### C.1.3.1 Parameter selection

In this example the public verification exponent $v$ is $2^{16}+1 = 65537$, which is odd. Therefore the secret prime factors $p$ and $q$ must satisfy:

$$\gcd(p-1, v) = \gcd(q-1, v) = 1.$$

$p$ = b843 ab40 c311 80cd 1063 f5d6 2158 bc6d 9c93
b2fc 1def 7dfd 3152 a695 89d9 4a80 1000 bfee fb62
7321 5552 2138 61d2 39a1.

$q$ = a8a1 c635 9063 d197 15a0 8e5f cd38 d6f2 3530
dde0 7359 2a67 1d02 e72a bb8e 8be2 599e 5bc8 ab53
c780 7d5e 9fd8 f680 5f79.

The public modulus $n$ is 767 bits long.

$n$ = 7960 d99c 1822 9f2c 2607 75d2 2eae 9941 6942
b3e8 f5ad c612 cd3f c529 70ed a698 0ba2 6388 08bd
b5cb c048 d63a 82d4 ac95 b166 9c43 4135 d7fc 19e2
022e a465 6bcc ee9b 7dba d90e 1125 91a2 1a66 1494
f4f6 e2b5 ce43 3b6a f390 79a0 2b48 c63f fc19.

$k_s = 766$.

The accreditation authority's private accreditation exponent $u$ is the least positive integer satisfying: $uv+1$ is a multiple of $\mathrm{lcm}(p-1, q-1)$.

$u$ = 02f8 d0c5 e0fe bd5a fd60 b80d 24c0 cdab d657
4b19 c1cf 8c1c 05be 86a5 ff1d 3289 0d2f e009 57fd
727d 6cab 3138 f989 c4e2 aa1d f4dd a901 f518 c560
7fa7 1ea7 4893 535e 1d2a a178 949a 0c25 cc59 25a6
25d6 337e 24c7 af7a 6fff 6d42 f61c d3ff 8dff.

$m = 1$. $t = 1$.

#### C.1.3.2 Identity selection

The identification data consists of $m = 1$ part. This identity part is constructed using the string 'Alex Ample' postfixed with a 16 bit field number.

$$I_{A1} \quad = \quad \texttt{416c 6578 2041 6d70 6c65 0001}$$

#### C.1.3.3 Accreditation generation

$J_{A1}$ = 3341 276c 2465 f078 5e20 9341 2a6d fe70 276c
2465 ee00 e301 9341 276c 2465 f078 5e20 9341 2a6d
fe70 276c 2465 ee00 e301 9341 276c 2465 f078 5e20
9341 2a6d fe70 276c 2465 ee00 e301 9141 276c 2465
f078 5e20 9341 2a6d fe70 276c 2465 ee00 e316.

$C_{A1}$ = 2c61 c981 f375 ed78 5a4e 9939 e054 c63a 3809
8f2f f525 ed20 2d4e 0a65 f7af 7548 80ce 954f 8f15
a1e0 bb73 9cbb 815c 5970 4f1c 4e3e 7552 dd1c 4966
d352 1992 149d f30c be32 d1c7 569b 40e4 8b7d b558
b003 95b8 2ec1 c1e6 3ed3 cfd9 abe0 22de 827c.

#### C.1.3.4 Authentication exchange

**Iteration 1:**

*Step (1):*

$r$ = 3a2c 36ef 335d b967 a76d b60f 7ad0 a6ea 518a
fcae 23d1 5ef3 3ead 463e 89af aa30 7a57 b5eb e1f5
b4aa 952e 125b 18be ac2f 245d f716 45e5 baee 9c5c
2750 4a76 92ac 0a45 bdd6 89a1 322f 6fa6 46d9 e18d
deb9 c948 e791 50cf 0776 104b 6f1d e369 977a.

$W$ = 22b6 a130 b77d 5ace 0eaf b6e6 d55f 3eab d710
cd08 184d df53 6cd1 0372 b0da 7f34 97cf 9355 87c1
c877 072d e166 c09f 6f3e 1b21 4952 6be0 774c e4fb
a38f f58b ddd6 5595 ec51 1e1f 7da2 5677 a22a 9e3d
6f45 23cf 1927 ac1e 76ff 6614 0d46 9f2d 44de.

*Step (3):*

$d_1$ = 003d.

*Step (5):*

$D$ = 1749 15ce a8cf cabb 678b 32d2 2424 c8fa 636e
cdca 7918 f47f b631 7741 e35e d84f 9257 bcd0 b8f6
27bc 7db4 b852 032b 8dea 028c 3681 a15c fedc ede8
8094 77c1 bec8 def4 c768 c78e 05f4 327e c58c c0f7
8229 8616 f15f 819b 746d 0efb 4747 9581 b39d.

*Step (7c):*

```
W'  =  22b6 a130 b77d 5ace 0eaf b6e6 d55f 3eab d710
cd08 184d df53 6cd1 0372 b0da 7f34 97cf 9355 87c1
c877 072d e166 c09f 6f3e 1b21 4952 6be0 774c e4fb
a38f f58b ddd6 5595 ec51 1e1f 7da2 5677 a22a 9e3d
6f45 23cf 1927 ac1e 76ff 6614 0d46 9f2d 44de.
```

## C.2 Mechanism based on discrete logarithms

### C.2.1 Example using 768-bit $p$, 128-bit $q$ and RIPEMD-128

#### C.2.1.1 Parameter selection

This example uses a 768-bit prime number $p$, a 128-bit prime number $q$ (a prime factor of $p-1$), and RIPEMD-128 as the hash-function.

The 768-bit prime number $p$:

$p$ = d716599e b22836ac fb221d0a f4c66b16 e3dceaee a73a17fb aaa33c07 6cf3571f 54d89d49 38d7c311 e24b98f1 e510599d b53f7387 0d2acf2f 8fbf8267 c1df4fe3 2e8a04e4 14125c5f d6d8efd7 8c5f1563 4288cd6a 0caaf4cd 3cf44434 d7ea8143.

The 128-bit prime number $q$, a prime factor of $(p-1)$[1]:

$q$ = a73a17fb aaa33c07 6cf3571f 54d89d49.

The element of order $q$ in $Z_p$:

$g$ = 5c7af3fa beff6338 f3137b85 a83e557b 49135e47 ba7ed438 e34b1fa0 af8c2651 15cf8b2f 3c924b33 0addf043 10ee6c41 a378541f 69a370dc b09f898a f3204864 8a8433be cdb55d5d 6cdbc85d b6f3a654 0df8b209 1a674d77 cdee3e1e 86d4fb93.

#### C.2.1.2 Key selection

The 128-bit private key of entity $A$:

$z_A$ = f1d4e85a eff74310 53adcac0 a9c155ce.

The 768-bit public verification key of entity $A$ ($y_A = g^{z_A} \bmod p$):

$y_A$ = 813eba80 bdbfdc78 15d09f74 ec21e1a6 4bd1a3c6 7a2591b2 44d53e2b 0c4dcb54 d5e2a8db 411e7c04 566d2671 12e72695 4a3c7699 a304255e fc58390c b0566240 66ee9d2b 131ebbca 7217f973 86dcab2b d4fb346c a2b5da9d 80954a65 4fa6f3c6.

#### C.2.1.3 Authentication exchange

*Step (1):*

Entity $A$ selects a random number $r$.

$r$ = 868b4b13 017364b7 e7dda29e cda55473.

$W = g^r \bmod p$
= 6ae62a6 d172be24 85830f5c c1524f4c a23172a8 c8691011 759f63d7 86b1a7d7 a6809f43 512f42c6 a6a444d6 a437f62f 02881f99 6e3638b0 93f6da41 cd4860e2 fb856e58 1a2cafed 8af85e6c 856ad852 c5f90648 915498bc d47fe84a 621c7bf3.

[1] Note that $p$ has been chosen so that a copy of $q$ is embedded within $p$. Such an approach to the choice of $p$ may be useful in situations where storage space and/or communications bandwidth is at a premium

*Step (2):*

$h(W)$ = f084606b 90de7902 2bb16d2f 31996976.

Entity $A$ sends $h(W)$ to entity $B$ (Token$AB_1 = h(W)$).

*Step (3):*

Entity $B$ chooses at random an integer $d$. (This example uses a 16-bit $d$.)

$d$ = d47c.

*Step (4):*

Entity $B$ sends challenge $d$ to entity $A$.

*Step (5):*

Entity $A$ computes the response $D$ as $D = r - dz_A \bmod q$.

$D$ = 1edee4bc 1667f13a 7cbf9d28 c5a356ea.

*Step (6):*

Entity $A$ sends Token$AB_2 = D$ to entity $B$.

*Step (7):*

Entity $B$ checks that $0 < D < q$, and calculates $W' = (y_A)^d g^D \bmod p$.

$(y_A)^d \bmod p$ = 6e6f703d ace31e6f 335f556b 42b24a6d 21771d60 2fa448be b74ae11d 3b63ff2f cecf2452 1417d470 04839f4a b15e91fa 72bbebfb 9b9c4b41 8c0c6a2d ef4a40e1 9a4a629c 32e63a0e fb03ec80 d55efeb8 173c6b26 58d28216 3be0ca6a 2d90cae6.

$g^D \bmod p$ = 8e249ba6 f0ef2f8a 156c0851 0bf23a2f 3408b9c0 7ec733cc dcfe78cf ca3bf160 7217be06 ccd9abc5 a392c7e7 4823070c 2e7c310a b26523a7 af58361c a685c3bb 1cd38f91 15479db1 cf38f7c8 852a4644 14cdc936 797e6883 7106bcb6 d1bbeaf1.

$W'$ = 6ae62a6 d172be24 85830f5c c1524f4c a23172a8 c8691011 759f63d7 86b1a7d7 a6809f43 512f42c6 a6a444d6 a437f62f 02881f99 6e3638b0 93f6da41 cd4860e2 fb856e58 1a2cafed 8af85e6c 856ad852 c5f90648 915498bc d47fe84a 621c7bf3.

If $h(W') = h(W)$, then authentication for entity $A$ is complete.

### C.2.2 Example using 1024-bit $p$, 160-bit $q$ and SHA-1

#### C.2.2.1 Parameter selection

This example uses a 1024-bit prime number $p$, a 160-bit prime number $q$ (a prime factor of $p-1$), and Dedicated

Hash-Function 3 (also known as SHA-1), specified in clause 9 of ISO/IEC 10118-3, as the hash-function.

The 1024-bit prime number $p$:

$p$ = ea9b8f92 26d7b2f6 729122ef 53ce81e2 567acf40 a7db660e ba5e4daf cb0ebc3a ccb15c36 896f67f0 703e7c69 afc4c24b 221a8968 5cdcfb3e 086d8f95 702cbfc5 8e4170a2 e10df7b5 2bf8f015 c5a689ca 48df291b e796c443 f5e7ad19 8c159f0a ba9d962e 60d34840 77b5993e 48bbc3ed fef5f54c accde46e 69a3f1f6 1ae08af9.

The 160-bit prime number $q$, a prime factor of $(p-1)$[1]:

$q$ = cb0ebc3a ccb15c36 896f67f0 703e7c69 afc4c24b.

The element of order $q$ in $Z_p$:

$g$ = 26324f69 934e6733 c66367a5 af5a08d8 455a5125 29882857 b20083e8 f72420a9 1f16a377 6dc612ff e652a2dd 05d51441 5f52c591 e8aa3127 8309ce2b ca9e5b73 5e8cc526 0dc1608d 91f32a8d 31265adc f2f2ff5f a4a786ef 25086bdb 061355cd 96ea33f6 429aef56 bc0c0aba db1ec3e0 b1140687 d60678c6 205c7f6d 6a236f87.

### C.2.2.2 Key selection

The 160-bit private key of entity $A$:

$z_A$ = 87146299 068b4b13 017364b7 e7dda29e cda5547e.

The 1024-bit public verification key of entity $A$ ($y_A = g^{z_A} \bmod p$):

$y_A$ = 819b36e6 62ddc4af 146dcf3a f888d61b 560ea5ea 8bb368f7 0e822e95 ef5e45c6 68b98732 725d29dc 21bf1394 29d95de2 98a6d595 9a7188c3 ab4b5d6d 20ca1d9e d6bc4d7a d23a4e3b 48cbe4ac da28d927 922c85ff db7e1f59 71a17dd5 dc68725c 32cf50f0 be5d8a73 f93bf113 1c55bf51 35b314be 5067fd31 9867041d 4c96e5cf.

### C.2.2.3 Authentication exchange

*Step (1):*

Entity $A$ selects a random number $r$.

$r$ = 87146299 068b4b13 017364b7 e7dda29e cda5547a.

$W = g^r \bmod p$
= 397ad6f9 b435b01b 4c43a2d1 008ddade 1a086c2f 0ea25134 ff5a8653 a374dfbf 47f1a543 fbb58232 0357cce1 33aeb861 6aebd4b7 65dea271 0dff3a09 7c40602b 7e719499 0e9c7717 0ce73286 930e9e27 f8053b28 d2c80fd2 ec529839 27f34f46 bb9842b0 bd9c6405 1b2c58d8 c5cdcc50 69c4a430 d0f93cd0 6f2f75f3 298684f6.

*Step (2):*

$h(W)$ = d3cf43cd 80f2525d 360bf266 d11590de 7efdb987.

Entity $A$ sends $h(W)$ to entity $B$ (Token$AB_1 = h(W)$).

*Step (3):*

Entity $B$ chooses at random an integer $d$. (This example uses a 40-bit $d$.)

$d$ = a2 cda554a6.

*Step (4):*

Entity $B$ sends the challenge $d$ to entity $A$.

*Step (5):*

Entity $A$ computes the response $D$ as $D = r - dz_A \bmod q$.

$D$ = 354bf25c 5f0e8cca f2aea2b9 7716a2d5 cb8ceb7e.

*Step (6):*

Entity $A$ sends Token$AB_2 = D$ to entity $B$.

*Step (7):*

Entity $B$ checks that $0 < D < q$, and calculates $W' = (y_A)^d g^D \bmod p$.

$(y_A)^d \bmod p$ = d95931d9 4ecd8e38 0993cf3d 9ab03767 abc0a08b 69a82166 83f73785 b940610f 9293ee53 be9e717f 6fd6a9be f7b0c140 1f374427 86856c96 c168f499 86800ecc 91f12765 be056ecb 7d03ce6b 4334a4d1 29cd1829 6705f4a6 105752c9 31190fe4 1a65c010 be4537f7 6913d471 50441aab 387a7e55 86e1debd 6343703f fd0eeef7.

$g^D \bmod p$ = c40924be 47db63b6 c48734a5 dd2f8a01 dc6c08ed 6cbfeda2 81b64230 fbbde7f8 fbddbd3e e64d6887 014b5b0a 78c0d111 c6550c01 01f00536 304bc91d 7efe0c1e f9dede7b 004534e0 74347241 b430ba21 bd1c2f93 903860b7 d1a14716 cc541c51 ade947ef 827e6a27 78d67db6 2b4db4ba 918ef0f8 7ccca628 f3042268 25988779.

$W'$ = 397ad6f9 b435b01b 4c43a2d1 008ddade 1a086c2f 0ea25134 ff5a8653 a374dfbf 47f1a543 fbb58232 0357cce1 33aeb861 6aebd4b7 65dea271 0dff3a09 7c40602b 7e719499 0e9c7717 0ce73286 930e9e27 f8053b28 d2c80fd2 ec529839 27f34f46 bb9842b0 bd9c6405 1b2c58d8 c5cdcc50 69c4a430 d0f93cd0 6f2f75f3 298684f6.

If $h(W') = h(W)$, then authentication for entity $A$ is complete.

[1] Note that $p$ has been chosen so that a copy of $q$ is embedded within $p$. Such an approach to the choice of $p$ may be useful in situations where storage space and/or communications bandwidth is at a premium

## C.3 Mechanism based on a trusted public transformation

### C.3.1 Example using 767-bit RSA and RIPEMD-160

#### C.3.1.1 Parameter selection

This example uses RSA as the asymmetric encipherment system, and Dedicated Hash-Function 1 (also known as RIPEMD-160), specified in clause 7 of ISO/IEC 10118-3, as the hash-function.

We suppose that $A$ uses a 767-bit RSA modulus $n = pq$, a public RSA exponent $e$, and a private RSA exponent $s$, where:

$p$ = cef2 8973 ddff 2ad1 ba38 4a98 71e0 7de1 d8ad 973f e2e1 2d6d 357c 19b2 7304 79b6 5c7e 6369 9a25 bb49 9f41 e7de 0f6f a105.

$q$ = 9327 da68 0aa9 a22f 201b 429a acf1 de30 382f cb01 cf3d 6b4b 85a1 fa3c f851 4738 5100 09ee 7dad 2b4c 4673 0971 a417 41ad.

$n$ = 76f5 7c6f 1d53 742a 45a2 1dab b9ca 6f4c eb1c 2317 6b02 9967 9ba4 3305 2c42 3146 44a3 9a79 5b57 5979 0685 8a10 932c f80d 8973 ecb6 30bf bc18 29db ff50 adf2 4465 a87c d236 1e8a 16c5 34f7 7acf ce94 0f59 234d c833 d279 0ade e26e 395f 71c5 1561.

$e$ = 4d97 cc58 6e72 3582 ae31 9d48 5814 05f5 4e74 1737 6710 f052 acda 8fd5 1827 b485 d762 c713 4bda b4bd 08d6 7f78 4a5c 174e d35b 5ff9 62eb 1965 6af3 a353 4635 e843 89a0 78f0 e042 e656 ff35 0236 33f4 cc86 fcbd 4349 7c0d 3c19 7d20 fc60 bc91 1017.

$s$ = 678a a11c 459c bd6d 10b9 9555 675a 7d7c 7850 af9b dc56 fe01 8846 7529 d02e 4aa6 85d0 3d2e 0e8e c027 ba69 3b4c f4dc 48c0 f2ad 74a2 25c4 fc38 ec52 94e4 a222 d922 7d53 389c c95f 9410 2b00 5840 247b ea8c 7a1f 3c60 ac3a 7297 704c 36e2 afbb e107.

$A$'s public encipherment transformation is $P_A(r) = r^e \pmod{n}$.

#### C.3.1.2 Authentication exchange

*Step (1):*

Entity $B$ chooses a random value $r$.

$r$ = adec c15b d356 b0cd 1b74 9469 b421 ac9d 28ee 96a5 85ec 6284 9cc2 8beb 0e59 4c9f 7377 f151 18fc a3df 249a b0aa ebb0 cf35 91f6 7858 d8a0 16e4 40bf ee20 bbcf da92 8e09 6e2a a6ec eccd f7f1.

$B$ computes $h(r)$, $r||h(r)$ and the challenge $d = P_A(r||h(r))$.

$h(r)$ = 4bc6 0e2c bfcb 238a 4d88 8b5f 3d4a eb02 3c16 1893.

$r||h(r)$ = adec c15b d356 b0cd 1b74 9469 b421 ac9d 28ee 96a5 85ec 6284 9cc2 8beb 0e59 4c9f 7377 f151 18fc a3df 249a b0aa ebb0 cf35 91f6 7858 d8a0 16e4 40bf ee20 bbcf da92 8e09 6e2a a6ec eccd f7f1 4bc6 0e2c bfcb 238a 4d88 8b5f 3d4a eb02 3c16 1893.

$d = P_A(r||h(r))$ = 60bd 94a5 7b0e 2eb9 0afb 5e21 630b 763f 8771 3479 98ed 4df6 3c9f bbf1 2369 d117 1c81 9277 63b9 3d5f 2af2 6288 7ee8 b24f b9d4 db2e c206 99b6 eb8f 2b31 3944 27e0 1f74 d501 cfca d45f 25ab dfd1 b605 c640 7d1a 597e 204a 1183 4670 c6b8 c52c 7cc3.

*Step (2):*

$B$ sends $d$ to $A$.

*Step (3):*

$A$ performs the following computational steps.

(a) $A$ recovers $r$ and $h(r)$ by calculating

$$r||h(r) = S_A(d) = d^s \pmod{n}.$$

(b) $A$ recomputes $h(r)$ from the value of $r$ recovered in the previous step, and compares it with the value of $h(r)$ recovered in the previous step.

Given that the values of $h(r)$ agree, $A$ sets $D = r$.

*Step (4):*

$A$ sends $D$ to $B$.

*Step (5):*

$B$ compares $D$ with $r$.

### C.3.2 Example using 1024-bit RSA and SHA-1

#### C.3.2.1 Parameter selection

This example uses RSA as the asymmetric encipherment system, and Dedicated Hash-Function 3 (also known as SHA-1), specified in clause 9 of ISO/IEC 10118-3, as the hash-function.

We suppose that $A$ uses a 1024-bit RSA modulus $n = pq$, a public RSA exponent $e$, and a private RSA exponent $s$, where:

$p$ = d329 dc1d 1156 1582 b2ec b9c3 90e8 5588 0e0e 5a6b 96b8 8e0f 8fe2 1a1c 6dd2 0d86 c85b 3932 1fdb f85d 713d aaae ad1c dab1 3571 c6d1 80a5 c0e7 7159 1be0 e4ad 54ad.

$q$ = bbb7 9564 0e4b 12d0 6c4a bfe3 1a93 f5f4 c69e 419f eac3 3c6c 2eaf e28e a60b eb6c 81c5 7477 1092 e8b2 67c3 6176 1969 cf37 1f26 60ad e102 6c5e c1c7 f394 6fa3 f72f.