INTERNATIONAL STANDARD

ISO/IEC 11770-4

Second edition
2017-11

AMENDMENT 2

Information technology — Security techniques — Key management —

Part 4:
Mechanisms based on weak secrets

AMENDMENT 2: Leakage-resilient password-authenticated key agreement with additional stored secrets

PROOF/ÉPREUVE

Reference number
ISO/IEC 11770-4:2017/Amd.2:2020(E)

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.

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) or the IEC list of patent declarations received (see https://patents.iec).

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 ISO/IEC JTC 1, *Information technology*, SC 27, *Information security, cybersecurity and privacy protection*.

A list of all parts in the ISO/IEC 11770 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.

Information technology — Security techniques — Key management —

Part 4: **Mechanisms based on weak secrets**

AMENDMENT 2: Leakage-resilient password-authenticated key agreement with additional stored secrets

Introduction

Insert new list item e) as follows:

e) **Leakage-resilient password-authenticated key agreement with additional stored secrets:** Establish one or more shared secret keys between two entities *A* and *B*, where *A* has a weak secret and a (possibly, insecure) stored secret that might be revealed to or altered by adversaries and *B* has verification data derived from *A*'s weak secret and stored secret. In a leakage-resilient password-authenticated key agreement with additional stored secrets mechanism, the shared secret keys are the result of a data exchange between the two entities; the shared secret keys are established if, and only if, the two entities have used the weak secret, the stored secret and the corresponding verification data; and *A*, *B* and an adversary who has obtained and altered the stored secret are all unable to predetermine the values of the shared secret keys.

NOTE 4 Here, "leakage-resilience" means security against either compromise of stored secrets held by client *A* or compromise of verification data held by server *B*, but not both. This type of key agreement mechanism is able to protect *A*'s weak secret from being discovered by *B*, as well as preventing an adversary from getting *A*'s weak secret from *B*. Also, this type of key agreement mechanism prevents an adversary from performing online dictionary attacks unless the adversary obtains *A*'s stored secret. In other words, an adversary who obtains *A*'s stored secret is restricted to performing online dictionary attacks, and the security level in this case is the same as that of the other mechanisms in this document. A typical application scenario would involve use between a client (*A*) and a server (*B*), where a client user employs a portable device such as a smart phone, USB memory or smart card to save the user's stored secret, or where a client terminal shares a network-attached storage device in an office environment.

Clause 2

Replace Clause 2 with the following:

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 10118 (all parts), *Information technology — Security techniques — Hash-functions*

ISO/IEC 29192-5, *Information technology — Security techniques — Lightweight cryptography — Part 5: Hash-functions*

ISO/IEC 9797 (all parts), *Information technology — Security techniques — Message Authentication Codes (MACs)*

ISO/IEC 29192-6, *IT Security techniques — Lightweight cryptography — Part 6: Message Authentication Codes (MACs)*

ISO/IEC 11770-6, *Information technology — Security techniques — Key management — Part 6: Key derivation*

ISO/IEC 18033-2, *Information technology — Security techniques — Encryption algorithms — Part 2: Asymmetric ciphers*

ISO/IEC 19772, *Information technology — Security techniques — Authenticated encryption*

Clause 3

Insert new term 3.40 as follows:

3.40
Hamming weight
number of non-zero elements in a bit string

Clause 4

Replace definitions as follows:

H	a collision-resistant hash-function taking an octet string as input and giving a bit string as output. One of the hash-functions specified in ISO/IEC 10118 (all parts) or ISO/IEC 29192-5 shall be used
$h(x, L_K)$	a collision-resistant hash-function taking an octet string x and an integer L_K as input and giving a bit string of length L_K (in bits) as output. One of the hash-functions specified in ISO/IEC 10118 (all parts) or ISO/IEC 29192-5 shall be used
$mac(k, m)$	a message authentication code (MAC) function taking a key k and a variable-length message m as input and giving a fixed-length output. One of the MAC algorithms specified in ISO/IEC 9797 (all parts) or ISO/IEC 29192-6 shall be used
G,G_a,G_b	points of order r on E over $F(q)$, where the relative discrete logarithms of G,G_a,G_b are unknown
g, g_1, g_a, g_b	elements of multiplicative order r in $F(q)$, where the relative discrete logarithms of g, g_1, g_a, g_b are unknown
K	a function for deriving a key from a secret value and a key derivation parameter. One of the key derivation functions specified in ISO/IEC 11770-6 shall be used

Add the following definitions:

$\oplus$	bit-wise exclusive-or operation on bit-strings of equal length
AE	an authenticated encryption, (reversible) transformation of data by a cryptographic algorithm to produce ciphertext that cannot be altered by an unauthorized entity without detection, i.e. that provides data confidentiality, data integrity, and data origin authentication. One of the authenticated encryption methods specified in ISO/IEC 19772 shall be used

Clause 5

Replace the last sentence with the following:

> It is also assumed that the entities are aware of a common hash-function H, one of the hash-functions specified in ISO/IEC 10118 (all parts) or ISO/IEC 29192-5.

Clause 9

Add new Clause 9 as follows:

9 Leakage-resilient password-authenticated key agreement with additional stored secrets

9.1 General

This clause specifies two mechanisms (LKAM1 and LKAM2) for leakage-resilient password-authenticated key agreement with additional stored secrets. These mechanisms, specified in 9.2 and 9.3, require one of the two entities to possess verification data for a weak secret and a stored secret known to the other entity.

The two mechanisms share the following initialization and key establishment processes.

Initialization process: The two entities involved agree to use a set of valid domain parameters, a set of key derivation parameters and a set of functions, all of which may be publicly known. The two entities also agree to use shared password-based information, i.e. one entity has a password-based weak secret and a stored secret, and the other entity has the corresponding verification data.

NOTE In the initialization process, the two entities who only share password-based information can establish a secure channel by performing the key establishment process. In the case of LKAM2, the server first sends an RSA parameter n. Through the established secure channel, the two entities can register verification data and stored secrets.

Key establishment process:

a) *Generate and exchange key tokens.* The two entities involved each randomly choose one or more key token factors associated with the domain parameters, create the corresponding key tokens, which may be associated with the password and the stored secret or verification data (a key token associated with the password and the stored secret or verification data is called an “entangled key token”), and then make the key tokens available to the other entity.

b) *Check validity of key tokens.* Depending on the operations for producing key tokens in step a), the two entities involved each choose an appropriate method to validate the received key tokens based on the domain parameters. If any validation fails, the entity involved shall output “invalid” and stop.

c) *Derive shared secret keys.* The two entities involved each apply certain secret value derivation functions to their own key token factor, the other entity's key tokens and/or shared verification data to produce a shared secret value. Each entity further applies a key derivation function to the shared secret value and the key derivation parameters, to derive one or more shared secret keys.

d) *Check key confirmation and update stored secrets.* The two entities involved use the shared secret keys established using the above steps to confirm their awareness of the keys to each other, and to update their stored secrets. This step is mandatory.

9.2 Leakage-resilient key agreement mechanism 1 (LKAM1)

9.2.1 General

This mechanism is designed to achieve leakage-resilient password-authenticated key agreement with additional stored secrets, and establishes one or more shared secret keys between entities A and B. In the mechanism, A has a password-based octet string π and a stored secret s_i, and B has verification data W_i corresponding to π and s_i. This mechanism provides unilateral explicit key authentication and, optionally, mutual key authentication.

This mechanism works in both the DL setting and the EC setting.

NOTE 1 In applications using leakage-resilient password-authenticated key agreement with additional stored secrets, A can play the role of a client and B can play the role of a server.

NOTE 2 This mechanism is based on the work of Shin, Kobara and Imai [35].

9.2.2 Prior shared parameters

Key agreement between two entities A and B takes place in an environment consisting of the following parameters:

— a set of valid domain parameters (either DL domain parameters or EC domain parameters) as specified in Clause 5;

— a counter which stores the value of i (initially $i = 1$);

— the length L_K of a shared secret key K;

— a password-based octet string π and a stored secret s_i, which is an integer of L_K bits used by A;

— a verification element derivation function J, used by A;

— a verification element $W_i = J(\pi, s_i)$, used by A and B;

— a key token generation function D, used by A and B;

— an entangled key token generation function C, used by A;

— a key token check function T, used by A and B;

— two secret value derivation functions V_A and V_B, one for each entity;

— a key derivation function K, used by A and B;

— one or more key derivation parameter octet strings $\{P_1, P_2, \ldots\}$, where A and B shall agree to use the same values.

9.2.3 Functions

9.2.3.1 Verification element derivation function *J*

The verification element derivation function J takes a password-based octet string π and a stored secret s_i as input and produces an element of $F(q)$, written $J(\pi, s_i)$, as output. Leakage-resilient key agreement mechanism 1 can be used with either of the following two functions J_{DL} and J_{EC}:

— J_{DL} is suitable for use when the mechanism is used with the DL domain parameters, i.e. it operates over the multiplicative group of elements of $F(q)$. Given the DL domain parameters (including g_b and q), a password-based octet string π and a stored secret s_i, J_{DL} is defined as in Formula (40):

$$J_{\mathrm{DL}}(\pi, s_i) = g_b^{\mathrm{BS2I}(H(\pi)) + s_i \bmod r} \bmod q \tag{40}$$

— J_{EC} is suitable for use when the mechanism is used with the EC domain parameters, i.e. it operates over the additive group of elements in an elliptic curve of $F(q)$. Given the EC domain parameters (including G_b), a password-based octet string π and a stored secret s_i, J_{EC} is defined as in Formula (41):

$$J_{\mathrm{EC}}(\pi, s_i) = [\mathrm{BS2I}(H(\pi)) + s_i \bmod r] \times G_b \tag{41}$$

Function BS2I (Bit String to Integer conversion) is described in Annex A.

9.2.3.2 Key token generation function *D*

The key token generation function D takes an integer x from $\{1, ..., r - 1\}$ as input, and produces an element written $D(x)$ as output. Leakage-Resilient Key Agreement Mechanism 1 can be used with either of the following two functions D_{DL} and D_{EC}:

— D_{DL} is suitable for use when the mechanism is used with the DL domain parameters, i.e. it operates over the multiplicative group of elements of $F(q)$. Given the DL domain parameters (including g and q) and an input x from $\{1, ..., r - 1\}$, D_{DL} is defined as in Formula (42):

$$D_{\mathrm{DL}}(x) = g^x \bmod q \tag{42}$$

— D_{EC} is suitable for use when the mechanism is used with the EC domain parameters, i.e. it operates over the additive group of elements in an elliptic curve of $F(q)$. Given the EC domain parameters (including G) and an input x from $\{1, ..., r - 1\}$, D_{EC} is defined as in Formula (43):

$$D_{\mathrm{EC}}(x) = [x] \times G \tag{43}$$

9.2.3.3 Entangled key token generation function *C*

The entangled key token generation function C takes two inputs, an output W_i of function J and an output X of function D, and produces an element written $C(W_i, X)$ as output. Leakage-Resilient Key Agreement Mechanism 1 can be used with either of the following C functions C_{DL} and C_{EC}:

— C_{DL} is suitable for use when the mechanism is used with the DL domain parameters, i.e. it operates over the multiplicative group of elements of $F(q)$. Given the DL domain parameters (including q) and two inputs, the output W_i of function J and the output X of function D, C_{DL} is defined as follows:

— compute $C_{\mathrm{DL}}(W_i, X) = W_i \times X \bmod q$;

— if $C_{\mathrm{DL}}(W_i, X)$ is 0, 1 or $q - 1$, output “invalid” and stop; otherwise, output $C_{\mathrm{DL}}(W_i, X)$.

— C_{EC} is suitable for use when the mechanism is used with the EC domain parameters, i.e. it operates over the additive group of elements in an elliptic curve of $F(q)$. Given the EC domain parameters and two inputs, the output W_i of function J and the output X of function D, C_{EC} is defined as follows:

— compute $C_{\mathrm{EC}}(W_i, X) = W_i + X$;

— if $[2^n] \times C_{\mathrm{EC}}(W_i, X) = 0_E$, output “invalid” and stop; otherwise output $C_{\mathrm{EC}}(W_i, X)$.

9.2.3.4 Key token check function *T*

The key token check function T is the same as that defined in 6.2.3.3.

9.2.3.5 Secret value derivation functions V_A and V_B

a) The secret value derivation function V_A takes two inputs, an integer x from $\{1, ..., r - 1\}$ and an output Y of function D, and produces an element written $V_A(x, Y)$ as output.

b) The secret value derivation function V_B takes three inputs, an integer y from {1, ..., $r - 1$}, an output X' of function C and an output W_i of function J, and produces an element written $V_B(y, X', W_i)$ as output.

c) V_A and V_B satisfy the condition $V_A(x, Y) = V_B(y, X', W_i)$.

Leakage-resilient key agreement mechanism 1 can be used with either of the following two functions V_{ADL} and V_{AEC}, and either of the following two functions V_{BDL} and V_{BEC}:

a) V_{ADL} is suitable for use when the mechanism is used with the DL domain parameters, i.e. it operates over the multiplicative group of $F(q)$. Given the DL domain parameters (including q), an integer x from {1, ..., $r - 1$} and an integer Y from {2, ..., $q - 2$}, V_{ADL} is defined in the following steps:

— compute $V_{ADL}(x, Y) = Y^x \bmod q$;

— output $V_{ADL}(x, Y)$.

b) V_{BDL} is suitable for use when the mechanism is used with the DL domain parameters, i.e. it operates over the multiplicative group of $F(q)$. Given the DL domain parameters (including q), an integer y from {1, ..., $r - 1$}, an integer X' from {2, ..., $q - 2$} and an integer W_i from {2, ..., $q - 2$}, V_{BDL} is defined in the following steps:

— compute $V_{BDL}(y, X', W_i) = (X' / W_i)^y \bmod q$;

— output $V_{BDL}(y, X', W_i)$.

c) V_{AEC} is suitable for use when the mechanism is used with the EC domain parameters, i.e. it operates over the additive group of elements in an elliptic curve of $F(q)$. Given the EC domain parameters, an integer x from {1, ..., $r - 1$} and a point $Y(\neq 0_E)$ on E, V_{AEC} is defined in the following steps:

— compute $V_{AEC}(x, Y) = [x] \times Y$;

— output $V_{AEC}(x, Y)$.

d) V_{BEC} is suitable for use when the mechanism is used with the EC domain parameters, i.e. it operates over the additive group of elements in an elliptic curve of $F(q)$. Given the EC domain parameters, an integer y from {1, ..., $r - 1$}, a point $X'(\neq 0_E)$ on E and a point $W_i(\neq 0_E)$ on E, V_{BEC} is defined in the following steps:

— compute $V_{BEC}(y, X', W_i) = [y] \times (X' - W_i)$;

— output $V_{BEC}(y, X', W_i)$.

9.2.3.6 Key derivation function *K*

The key derivation function K is the same as that defined in 6.2.3.6.

9.2.4 Initialization operation

In the initialization operation, A chooses an integer s_1 randomly from {1, ..., $r - 1$}, computes $W_1 = J(\pi, s_1)$, and then securely transfers W_1 to B. While A has the password-based weak secret π and the stored secret s_1 (along with the counter $i = 1$), B has the corresponding verification data W_1 and the counter $i = 1$.

NOTE Entity A can update the password-based weak secret by performing the initialization operation.

9.2.5 Key agreement operation

In the i-th ($i \geq 1$) key agreement operation, this mechanism involves both A and B performing a sequence of up to three steps, numbered A1 to A3 and B1 to B3 (for the steps to be followed by A and B, respectively).

a) Entangled key token construction (A1)

A performs the following steps:

1) compute $W_i = J(\pi, s_i)$ as its verification element;

2) choose an integer x randomly from $\{1, ..., r - 1\}$ as its key token factor;

3) compute $X = D(x)$ as its key token, and $X' = C(W_i, X)$ as its entangled key token (if the output of function C is "invalid", go back to the above item to choose a different x value at random and try again);

4) make i and X' available to B.

b) Key token construction (B1)

B performs the following steps:

1) receive i and X' from A;

2) check the validity of i: if the received value of i is not the same as the value B has, output "invalid" and stop; otherwise, continue;

3) check the validity of X' using $T(X')$: if $T(X') = 0$, output "invalid" and stop; otherwise, continue;

4) choose an integer y uniformly at random from the range $\{1, ..., r - 1\}$ as its key token factor;

5) compute $Y = D(y)$ as its key token;

6) compute $z = V_B(y, X', W_i)$ as an agreed secret value;

7) compute $o_B = H(\mathrm{I2OS}(1)||A||B||\mathrm{I2OS}(i)||\mathrm{GE2OS}_X(X')||\ \mathrm{GE2OS}_X(Y)||\ \mathrm{GE2OS}_X(W_i)||\ \mathrm{GE2OS}_X(z))$;

8) make Y and o_B available to A.

c) Key confirmation (mandatory) and shared secret key derivation (A2)

A performs the following steps:

1) receive Y and o_B from B;

2) check the validity of Y using $T(Y)$: if $T(Y) = 0$, output "invalid" and stop; otherwise, continue;

3) compute $z = V_A(x, Y)$ as an agreed secret value;

4) compute $o_B' = H(\mathrm{I2OS}(1)||A||B||\mathrm{I2OS}(i)||\mathrm{GE2OS}_X(X')||\ \mathrm{GE2OS}_X(Y)||\ \mathrm{GE2OS}_X(W_i)||\ \mathrm{GE2OS}_X(z))$;

5) if $o_B \neq o_B'$, output "invalid" and stop; otherwise, continue;

6) compute $o_A = H(\mathrm{I2OS}(2)||A||B||\mathrm{I2OS}(i)||\mathrm{GE2OS}_X(X')||\ \mathrm{GE2OS}_X(Y)||\ \mathrm{GE2OS}_X(W_i)||\ \mathrm{GE2OS}_X(z))$;

7) compute $K_j = K(A||B||\mathrm{I2OS}(i)||\mathrm{GE2OS}_X(X')||\ \mathrm{GE2OS}_X(Y)||\ \mathrm{GE2OS}_X(W_i)||\ \mathrm{GE2OS}_X(z), P_j, L_K)$ as a shared secret key for each key derivation parameter P_j ($j = 1, 2, ...$);

8) make o_A available to B.

d) Key confirmation and shared secret key derivation (B2)

B performs the following steps:

1) receive o_A from A;

2) compute $o_A' = H(\mathrm{I2OS}(2)||A||B||\mathrm{I2OS}(i)||\mathrm{GE2OS}_X(X')||\ \mathrm{GE2OS}_X(Y)||\ \mathrm{GE2OS}_X(W_i)||\ \mathrm{GE2OS}_X(z))$;

3) if $o_A \neq o_A'$, output "invalid" and stop; otherwise, continue;

4) compute $K_j = K(A||B||\mathrm{I2OS}(i)||\mathrm{GE2OS}_X(X')||\ \mathrm{GE2OS}_X(Y)||\ \mathrm{GE2OS}_X(W_i)||\ \mathrm{GE2OS}_X(z), P_j, L_K)$ as a shared secret key for each key derivation parameter P_j ($j = 1, 2, ...$).

e) Update of stored secrets (A3 and B3)

A performs the following step (A3):

1) compute $s_{(i+1)} = s_i + \mathrm{BS2I}(H(\mathrm{I2OS}(3)||A||B||\mathrm{I2OS}(i)||\mathrm{GE2OS}_X(X')||\ \mathrm{GE2OS}_X(Y)||\ \mathrm{GE2OS}_X(W_i)||\ \mathrm{GE2OS}_X(z)))\bmod r$.

B performs the following step (B3):

1) compute $u = \mathrm{BS2I}(H(\mathrm{I2OS}(3)||A||B||\mathrm{I2OS}(i)||\mathrm{GE2OS}_X(X')||\ \mathrm{GE2OS}_X(Y)||\ \mathrm{GE2OS}_X(W_i)||\ \mathrm{GE2OS}_X(z)))\bmod r$;

2) compute $W_{(i+1)} = W_i \times (g_b)^u \bmod q$ in the DL setting;

3) compute $W_{(i+1)} = W_i + [\mathrm{u}] \times G_b$ in the EC setting;

4) check the validity of $W_{(i+1)}$ using $T(W_{(i+1)})$: if $T(W_{(i+1)}) = 0$, output "invalid" and stop.

Entity *A* shall verify the entity *B*'s proof of knowledge of the agreed key before revealing any information derived from the agreed key. Therefore, A2 shall be done before B2 if the latter is performed.

Functions BS2I (Bit String to Integer conversion), I2OS (Integer to Octet String conversion) and $\mathrm{GE2OS}_X$ (Group Element to Octet String conversion) are described in Annex A where Annex A shall be referenced for the details of the conversion functions.

Numerical examples can be found in Annex D.1.

NOTE 1 A group element in this mechanism is a point on the curve *E* in the EC setting, or an integer in the range $\{1, ..., q - 1\}$ in the DL setting.

NOTE 2 In this mechanism, *X* and *Y* can be computed before the key agreement operation.

NOTE 3 This mechanism can be extended to provide synchronization, randomized ID and security against server compromise impersonation attacks in the same way as in Section 6 of Reference [36].

9.3 Leakage-resilient key agreement mechanism 2 (LKAM2)

9.3.1 General

This mechanism is designed to achieve leakage-resilient authenticated key agreement with password and untrusted storage using the RSA public key cryptosystem and establishes one or more shared secret keys between entities *A* and *B* with joint key control. In the mechanism, *A* remembers a password (denoted byπ when rendered as an octet string), and has, in an untrusted storage device that might be modified or copied by adversaries to impersonate *A* or *B* or to reveal the agreed keys, the RSA parameter *n*, a stored secret u_j where subscript *j* denotes a counter and a random pseudo identity *A*′ of *A*. *B* has password verification data v_j corresponding to both π and u_j, a hash value *A*′′ of *A*′, and RSA private key parameters *d*, *p*, *q* and so on. These RSA private and public key parameters shall be generated using ISO/IEC 18033-2:2006, 11.1, and the additional requirement and recommendations for *e* to be used in this mechanism are explained in 9.3.3. This mechanism provides unilateral explicit key authentication, and optionally mutual key authentication.

NOTE 1 In applications using leakage-resilient authenticated key agreement with password and untrusted storage, *A* can play the role of a client and *B* can play the role of a server.

NOTE 2 This mechanism is based on the work of Shin, Kobara and Imai [36].

9.3.2 Prior shared parameters

Key agreement between two entities *A* and *B* takes place in an environment consisting of the following parameters, in which L_K, *e*, *H*, *mac*, *K*, $\{P_1, P_2, ...\}$ are shared as system parameters among all or a subset of the users of this mechanism:

— the length of a shared secret key L_K which must be a multiple of 8;

— a set of RSA public key parameters, namely a prime integer *e* and a composite number *n* generated as specified in ISO/IEC 18033-2:2006, 11.1, where *e* is specialized for this mechanism as explained in 9.3.3;

— a cryptographic collision-resistant hash-function *H* giving a $2L_K$-bit output, that shall be chosen from amongst the functions standardized in ISO/IEC 10118 (all parts) or ISO/IEC 29192-5, being truncated as necessary;

— a message authentication code generation function *mac*, that shall be chosen from amongst the functions standardized in ISO/IEC 9797 (all parts) or ISO/IEC 29192-6;

— a counter which stores the value of *j* (initially *j* =1);

— a stored secret u_j and corresponding $v_j = J(\pi,u_j)$ where u_j is an octet string of random L_K bits and *J* is a password verification element derivation function in 9.3.4.1. Both u_j and v_j are set in the initialization operation in 9.3.5, and then used by *A* and *B*, respectively;

— a pseudo identity A'_j and a corresponding hash value $A''_j = H(\text{I2OS}(0)||A'_j)$ of the entity *A*, where A'_j is an octet string of length $L_K/8$ whose constituent bits are generated uniformly at random. Both A'_j and A''_j are set in the initialization operation in 9.3.5 and used by *A* and *B*, respectively;

— a key derivation function *K*, that shall be chosen from amongst the functions standardized in ISO/IEC 11770-6;

— one or more key derivation parameter octet strings $\{P_1, P_2, ...\}$, where *A* and *B* shall agree to use the same values.

9.3.3 Additional requirement and recommendations for RSA public key parameter *e*

The following requirements on the choice of the parameter *e*, additional to those specified in ISO/IEC 18033-2:2006, 11.1, apply.

a) *e* shall be a prime and $e \geq 2^{L_K}$;

b) the sum of the Hamming weight of the binary representation of *e* and the bitlength of *e* should be as small as possible, subject to requirement a);

c) *e* should be as large as possible within b).

NOTE 1 Requirement a) ensures that the *e*-residue attack [36], which applies when an adversary can modify the parameter *n*, is not feasible.

NOTE 2 Requirement b) is intended to help minimize the computational cost for entity *A*.

NOTE 3 Requirement c) is intended to make *e*-residue attacks as difficult as possible for a given computational cost on *A*.

NOTE 4 Examples of choices for *e* satisfying requirement a)-c) are given in C.4.

9.3.4 Functions

9.3.4.1 Password verification element derivation function *J*

The password verification element derivation function *J* takes a password-based octet string π and a random L_K-bit stored secret u_j as input, and produces as output an octet string password verification data $v_j = H(\text{I2OS}(4)||\pi||A||B) \oplus u_j$.

9.3.4.2 Key token generation function *D*