INTERNATIONAL STANDARD

ISO/IEC 11770-3

First edition
1999-11-01

Information technology — Security techniques — Key management —

Part 3:
Mechanisms using asymmetric techniques

Technologies de l'information — Techniques de sécurité — Gestion de clés —

Partie 3: Mécanismes utilisant des techniques asymétriques

Reference number
ISO/IEC 11770-3:1999(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 734 10 79
E-mail copyright@iso.ch
Web www.iso.ch

Printed in Switzerland

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 3.

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. 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.

Attention is drawn to the possibility that some of the elements of this part of ISO/IEC 11770 may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

International Standard ISO/IEC 11770-3 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT Security techniques*.

ISO/IEC 11770 consists of the following parts, under the general title *Information technology — Security techniques — Key management*:

— *Part 1: Framework*

— *Part 2: Mechanisms using symmetric techniques*

— *Part 3: Mecanisms using asymmetruc techniques*

Further parts may follow.

Annexes A to E of this part of ISO/IEC 11770 are for information only.

Information technology — Security techniques — Key management —

Part 3:
Mechanisms using asymmetric techniques

1. Scope

This part of ISO/IEC 11770 defines key management mechanisms based on asymmetric cryptographic techniques. It specifically addresses the use of asymmetric techniques to achieve the following goals:

1. Establish a shared secret key for a symmetric cryptographic technique between two entities *A* and *B* by key agreement. In a secret key agreement mechanism the secret key is the result of a data exchange between the two entities *A* and *B*. Neither of them can predetermine the value of the shared secret key.
2. Establish a shared secret key for a symmetric cryptographic technique between two entities *A* and *B* by key transport. In a secret key transport mechanism the secret key is chosen by one entity *A* and is transferred to another entity *B*, suitably protected by asymmetric techniques.
3. Make an entity's public key available to other entities by key transport. In a public key transport mechanism, the public key of an entity A must be transferred to other entities in an authenticated way, but not requiring secrecy.

Some of the mechanisms of this part of ISO/IEC 11770 are based on the corresponding authentication mechanisms in ISO/IEC 9798-3.

This part of ISO/IEC 11770 does not cover aspects of key management such as

- key lifecycle management,
- mechanisms to generate or validate asymmetric key pairs,
- mechanisms to store, archive, delete, destroy, etc. keys.

While this part of ISO/IEC 11770 does not explicitly cover the distribution of an entity's private key (of an asymmetric key pair) from a trusted third party to a requesting entity, the key transport mechanisms described can be used to achieve this.

This part of ISO/IEC 11770 does not cover the implementations of the transformations used in the key management mechanisms.

NOTE - To achieve authenticity of key management messages it is possible to make provisions for authenticity within the key establishment protocol or to use a public key signature system to sign the key exchange messages.

2. Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of ISO 11770. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of ISO 11770 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 7498-2:1989, *Information processing systems - Open Systems Interconnection - Basic Reference Model - Part 2: Security Architecture*.

ISO/IEC 9594-8:1995, *Information technology - Open Systems Interconnection – The Directory: Authentication framework.*

ISO/IEC 9798-3:1998, *Information technology - Security techniques - Entity authentication - Part 3: Mechanisms using digital signature techniques.*

ISO/IEC 10118-1:1994, *Information technology - Security techniques - Hash-functions - Part 1: General*.

ISO/IEC 10181-1:1996, *Information technology - Open Systems Interconnection - Security frameworks for open systems Overview.*

ISO/IEC 11770-1:1996, *Information technology - Security techniques - Key management - Part 1: Framework*.

3. Definitions

For the purposes of this part of ISO/IEC 11770, the following definitions apply.

3.1. asymmetric cryptographic technique: a cryptographic technique that uses two related transformations, a public transformation (defined by the public key) and a private transformation (defined by the private key). The two transformations have the property that, given the public transformation, it is computationally infeasible to derive the private transformation.

NOTE - A system based on asymmetric cryptographic techniques can either be an encipherment system, a signature system, a combined encipherment and signature system, or a key agreement system. With asymmetric cryptographic techniques there are four elementary transformations: sign and verify for signature systems, encipher and decipher for encipherment systems. The signature and the decipherment transformation are kept private by the owning entity, whereas the corresponding verification and encipherment transformations are published. There exist asymmetric cryptosystems (e.g. RSA) where the four elementary functions may be achieved by only two transformations: one private transformation suffices for both signing and decrypting messages, and one public transformation suffices for both verifying and encrypting messages. However, since this does not conform to the principle of key separation, throughout this part of ISO/IEC 11770 the four elementary transformations and the corresponding keys are kept separate.

3.2. asymmetric encipherment system: a system based on asymmetric cryptographic techniques whose public transformation is used for encipherment and whose private transformation is used for decipherment.

3.3. asymmetric key pair: a pair of related keys where the private key defines the private transformation and the public key defines the public transformation.

3.4. certification authority (CA): a center trusted to create and assign public key certificates. Optionally, the certification authority may create and assign keys to the entities.

3.5. cryptographic check function: a cryptographic transformation which takes as input a secret key and an arbitrary string, and which gives a cryptographic check value as output. The computation of a correct check value without knowledge of the secret key shall be infeasible [ISO/IEC 9798-1:1997].

3.6. cryptographic check value: information which is derived by performing a cryptographic transformation on the data unit [ISO/IEC 9798-4:1995].

3.7. decipherment: the reversal of a corresponding encipherment [ISO/IEC 11770-1:1996].

3.8. digital signature: a data appended to, or a cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the origin and integrity of the data unit and protect the sender and the recipient of the data unit against forgery by third parties, and the sender against forgery by the recipient.

3.9. distinguishing identifier: information which unambiguously distinguishes an entity [ISO/IEC 11770-1:1996].

3.10. encipherment: the (reversible) transformation of data by a cryptographic algorithm to produce ciphertext, i.e. to hide the information content of the data [ISO/IEC 11770-1:1996].

3.11. entity authentication: the corroboration that an entity is the one claimed [ISO/IEC 9798-1:1997].

3.12. entity authentication of A to B: the assurance of the identity of entity A for entity B.

3.13. explicit key authentication from A to B: the assurance for entity B that A is the only other entity that is in possession of the correct key.

NOTE - implicit key authentication from A to B and key confirmation from A to B together imply explicit key authentication from A to B.

3.14. implicit key authentication from A to B: the assurance for entity B that A is the only other entity that can possibly be in possession of the correct key.

3.15. key: a sequence of symbols that controls the operation of a cryptographic transformation (e.g. encipherment, decipherment, cryptographic check function computation, signature calculation, or signature verification) [ISO/IEC 11770-1:1996].

3.16. key agreement: the process of establishing a shared secret key between entities in such a way that neither of them can predetermine the value of that key.

3.17. key confirmation from A to B: the assurance for entity B that entity A is in possession of the correct key.

3.18. key control: the ability to choose the key or the parameters used in the key computation.

3.19. key establishment: the process of making available a shared secret key to one or more entities. Key establishment includes key agreement and key transport.

3.20. key token: key management message sent from one entity to another entity during the execution of a key management mechanism.

3.21. key transport: the process of transferring a key from one entity to another entity, suitably protected.

3.22. mutual entity authentication: entity authentication which provides both entities with assurance of each other's identity.

3.23. one-way function : a function with the property that it is easy to compute the output for a given input but it is computationally infeasible to find for a given output an input which maps to this output.

3.24. private key: that key of an entity's asymmetric key pair which can only be used by that entity.

NOTE - In the case of an asymmetric signature system the private key defines the signature transformation. In the case of an asymmetric encipherment system the private key defines the decipherment transformation.

3.25. public key that key of an entity's asymmetric key pair which can be made public

NOTE - In the case of an asymmetric signature system the public key defines the verification transformation. In the case of an asymmetric encipherment system the public key defines the encipherment transformation. A key that is 'publicly known' is not necessarily globally available. The key may only be available to all members of a pre-specified group.

3.26. public key certificate the public key information of an entity signed by the certification authority and thereby rendered unforgeable.

3.27. public key information: information containing at least the entity's distinguishing identifier and public key. The public key information is limited to data regarding one entity, and one public key for this entity. There may be other static information regarding the certification authority, the entity, the public key, restrictions on key usage, the validity period, or the involved algorithms, included in the public key information.

3.28. secret key: a key used with symmetric cryptographic techniques by a specified set of entities.

3.29. sequence number: a time variant parameter whose value is taken from a specified sequence which is non-repeating within a certain time period [ISO/IEC 11770-1:1996].

3.30. signature system: a system based on asymmetric cryptographic techniques whose private transformation is used for signing and whose public transformation is used for verification.

3.31. time stamp: a data item which denotes a point in time with respect to a common time reference.

3.32. time stamping authority: a trusted third party trusted to provide evidence which includes the time when the secure time stamp is generated [ISO/IEC 13888-1:1997].

3.33. time variant parameter: a data item used to verify that a message is not a replay, such as a random number, a sequence number, or a time stamp.

3.34. trusted third party: a security authority, or its agent, trusted by other entities with respect to security related activities [ISO/IEC 10181-1:1996].

4. Symbols and abbreviations

The following symbols and abbreviations are used in this part of ISO/IEC 11770.

A,B	distinguishing identifiers of entities.
BE	enciphered data block
BS	signed data block
CA	certification authority.
$Cert_A$	entity *A*'s public key certificate
D_A	entity *A*'s private decipherment transformation.
d_A	entity *A*'s private decipherment key.
E_A	entity *A*'s public encipherment transformation.
e_A	entity *A*'s public encipherment key.
F(h,g)	the key agreement function.
f	cryptographic check function
$f_K(Z)$	cryptographic check value which is the result of applying the cryptographic check function *f* using as input a secret key *K* and an arbitrary data string *Z*.
g	the common element shared publicly by all the entities that use the key agreement function *F*.
h_A	entity *A*'s private key agreement key.
hash	hash-function
H	set of elements
G	set of elements
K	a secret key for a symmetric cryptosystem.
K_{AB}	a secret key shared between entities *A* and *B*.

NOTE - In practical implementations the shared secret key may be subject to further processing before it can be used for a symmetric cryptosystem.

KT	key token.
KT_{Ai}	the key token sent by entity *A* after processing phase i.
p_A	entity *A*'s public key agreement key.
PKI_A	entity *A*'s public key information
r	a random number generated in the course of a mechanism.
r_A	a random number issued by entity *A* in a key agreement mechanism.
S_A	entity *A*'s private signature transformation.
s_A	entity *A*'s private signature key.
Texti	an optional data field whose use is beyond the scope of this part of ISO/IEC 11770.
TVP	time-variant parameter, such as a random number, a time stamp, or a sequence number.
V_A	entity *A*'s public verification transformation.
v_A	entity *A*'s public verification key.
w	one-way function
Σ	the digital signature
$\|\|$	concatenation of two data elements.

NOTES

1. No assumption is made on the nature of the signature transformation. In the case of a signature system with message recovery, $S_A(m)$ denotes the signature Σ itself. In the case of a signature system with appendix, $S_A(m)$ denotes the message *m* together with signature Σ.

2. The keys of an asymmetric cryptosystem are denoted by a lower case letter (indicating the function of that key) indexed with the identifier of its owner, e.g. the public verification key of entity *A* is denoted by v_A. The corresponding transformations are denoted by upper case letters indexed with the identifier of their owner, e.g. the public verification transformation of entity *A* is denoted by V_A.

5. Requirements

It is assumed that the entities are aware of each other’s claimed identities. This may be achieved by the inclusion of identifiers in information exchanged between the two entities, or it may be apparent from the context of the use of the mechanism. Verifying the identity means to check that a received identifier field agrees with some known (trusted) value or prior expectation.

If a public key is registered with an entity then that entity shall make sure that the entity who registers the key is in possession of the corresponding private key (see Part 1 for registration of key).

6. Secret key agreement

Key agreement is the process of establishing a shared secret key between two entities A and B in such a way that neither of them can predetermine the value of the shared secret key. Key agreement mechanisms may provide for implicit key authentication; in the context of key establishment, implicit key authentication means that after the execution of the mechanism only an identified entity can be in possession of the correct shared secret key.

The key agreement between two entities A and B takes place in a context shared by the two entities. The context consists of the following objects: a set G, a set H and a function F. The function F shall satisfy the following requirements:

1. F operates on two inputs, one element h from H and one element g from G, and produces a result y in G, $y = F(h,g)$.
2. F satisfies the commutativity condition $F(h_A,F(h_B,g)) = F(h_B,F(h_A,g))$.
3. It is computationally intractable to find $F(h_1,F(h_2,g))$ from $F(h_1,g)$, $F(h_2,g)$ and g. This implies that $F(\cdot,g)$ is a one-way function.
4. The entities A and B share a common element g in G which may be publicly known.
5. The entities acting on this setting can efficiently compute function values $F(h,g)$ and can efficiently generate random elements in H.

Depending on the particular key agreement mechanism further conditions may be imposed.

NOTES

1. An example of a possible function F is given in Annex B.
2. In practical implementations of the key agreement mechanisms the shared secret key may be subject to further processing. A derived shared secret key may be computed (1) by extracting bits from the shared secret key K_{AB} directly or (2) by passing the shared secret K_{AB} and optionally other nonsecret data through a one-way function and extracting bits from the output.
3. It will in general be necessary to check the received function values $F(h,g)$ for weak values. If such values are encountered, the protocol shall be aborted. An example known as Diffie-Hellman key agreement is given in clause B.5.

6.1 Key agreement mechanism 1

This key agreement mechanism non-interactively establishes a shared secret key between entities A and B with mutual implicit key authentication. The following requirements shall be satisfied:

1. Each entity X has a private key agreement key h_X in H and a public key agreement key $p_X = F(h_X,g)$.
2. Each entity has access to an authenticated copy of the public key agreement key of the other entity. This may be achieved using the mechanisms of clause 8.

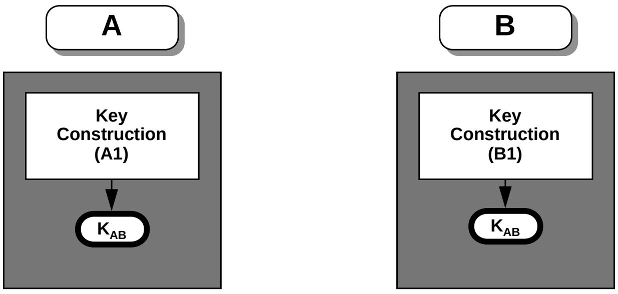

Figure 1 - Key Agreement Mechanism 1

Key Construction (A1) A computes, using its own private key agreement key h_A and B's public key agreement key p_B, the shared secret key as

$$K_{AB} = F(h_A, p_B)$$

Key Construction (B1) B computes, using its own private key agreement key h_B and A's public key agreement key p_A, the shared secret key as

$$K_{AB} = F(h_B, p_A)$$

As a consequence of requirement 2 of *F*, the two computed values for the key K_{AB} are identical.

NOTE - This Key Agreement Mechanism has the following properties:

1. Number of passes: 0. As a consequence, the secret shared key has always the same value (but see clause 6 note 2).

2. Key authentication: this mechanism provides mutual implicit key authentication.

3. Key confirmation: this mechanism provides no key confirmation.

4. This is a key agreement mechanism since the established key is a one-way function of the private key agreement keys h_A and h_B of *A* and *B* respectively. However, one entity may know the other entity's public key prior to choosing their private key. Such an entity may select approximately *s* bits of the established key, at the cost of generating 2^s candidate values for their private key agreement key in the interval between discovering the other entity's public key and choosing their own private key.

5. Example: an example known as Diffie-Hellman key agreement is given in clause B.5.

6.2 Key agreement mechanism 2

This key agreement mechanism establishes in one pass a shared secret key between *A* and *B* with implicit key authentication from *B* to *A*, but no entity authentication from *A* to *B* (i.e. *B* does not know with whom it has established the shared secret key). The following requirements shall be satisfied:

1. Entity *B* has a private key agreement key h_B in *H* and a public key agreement key $p_B = F(h_B, g)$.

2. Entity *A* has access to an authenticated copy of *B*'s public key agreement key p_B. This may be achieved using the mechanisms of clause 8.

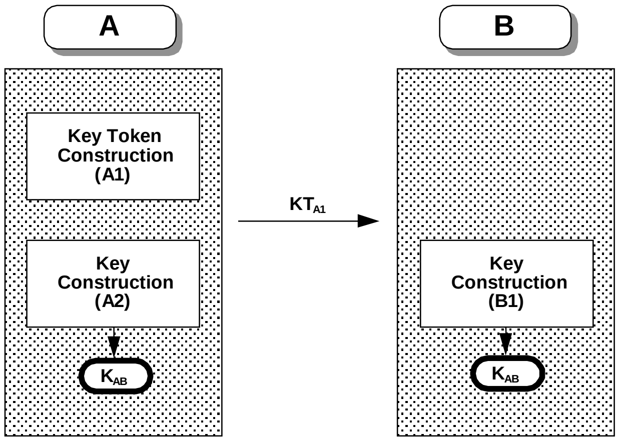

Figure 2 - Key Agreement Mechanism 2

Key Token Construction (A1) *A* randomly and secretly generates *r* in *H*, computes $F(r,g)$ and sends the key token

$$KT_{A1} = F(r,g) \,||\, Text$$

to *B*.

Key Construction (A2) Further *A* computes the key as

$$K_{AB} = F(r, p_B)$$

Key Construction (B1) *B* extracts $F(r,g)$ from the received key token KT_{A1} and computes the shared secret key

$$K_{AB} = F(h_B, F(r,g))$$

According to requirement 2 of *F*, the two computed values for the key K_{AB} are identical.

NOTE - This Key Agreement Mechanism has the following properties:

1. Number of passes: 1.

2. Key authentication: this mechanism provides implicit key authentication from *B* to *A* (*B* is the only entity other than *A* who can compute the shared secret key).

3. Key confirmation: this mechanism provides no key confirmation.

4. This is a key agreement mechanism since the established key is a one-way function of a random value *r* supplied by *A* and *B*'s private key agreement key. However, since entity *A* may know entity *B*'s public key prior to choosing the value *r*,

entity A may select approximately s bits of the established key, at the cost of generating 2^s candidate values for r in the interval between discovering B's public key and sending KT_{A1}.

5. Example: an example of this key agreement mechanism (known as ElGamal key agreement) is described in clause B.3.

6. Key usage: as B receives the key K_{AB} from the non-authenticated entity A, secure usage of K_{AB} at B's end is restricted to functions not requiring trust in A's authenticity such as decipherment and generation of message authentication codes.

6.3 Key agreement mechanism 3

This key agreement mechanism establishes in one pass a shared secret key between A and B with mutual implicit key authentication, and entity authentication of A to B. The following requirements shall be satisfied:

1. Entity A has an asymmetric signature system (S_A,V_A).
2. Entity B has access to an authenticated copy of the public verification transformation V_A. This may be achieved using the mechanisms of clause 8.
3. Entity B has a key agreement system with keys (h_B, p_B).
4. Entity A has access to an authenticated copy of the public key agreement key p_B of entity B. This may be achieved using the mechanisms of clause 8.
5. *TVP:* The *TVP* shall either be a time stamp or a sequence number. If time stamps are used, secure and synchronized time clocks are required; if sequence numbers are used, the ability to maintain and verify bilateral counters is required.
6. The entities A and B have agreed on a cryptographic check function f (such as those standardized in ISO/IEC 9797) and a way to incorporate K_{AB} as the key in this check function.

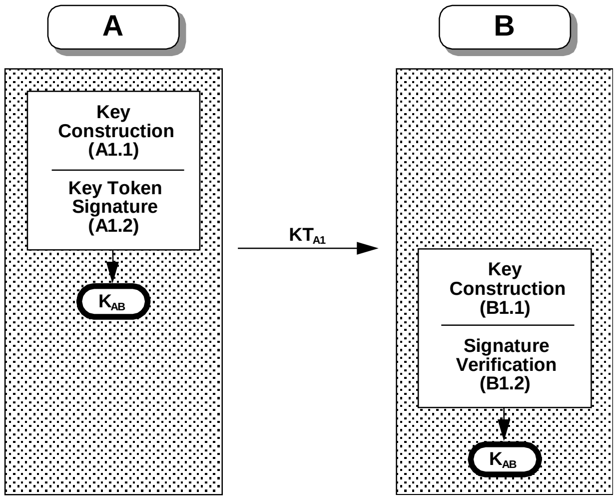

Figure 3 - Key Agreement Mechanism 3

Key Construction (A1.1) A randomly and secretly generates r in and computes $F(r,g)$. A computes the shared secret key as

$$K_{AB} = F(r,p_B)$$

Using the shared secret key K_{AB}. A computes a cryptographic check value on the concatenation of the sender's distinguishing identifier A and a sequence number or time stamp *TVP*.

Key Token Signature (A1.2) A signs the cryptographic check value, using its private signature transformation S_A. Then A forms the key token, consisting of the sender's distinguishing identifier A, the key input $F(r,g)$, the *TVP*, the signed cryptographic check value, and some optional data

$$KT_{A1} = A||F(r,g)||TVP|| S_A(f_{K_{AB}}(A||TVP))||Text1$$

and sends it to B.

Key Construction (B1.1) B extracts $F(r,g)$ from the received key token and computes the shared secret key, using its private key agreement key h_B,

$$K_{AB} = F(h_B,F(r,g))$$

Using the shared secret key K_{AB} B computes the cryptographic check value on the sender's distinguishing identifier A and the *TVP*.

Signature Verification (B1.2) *B* uses the sender's public verification transformation V_A to verify *A*'s signature and thus the integrity and origin of the received key token KT_{A1}. Then *B* validates the timeliness of the token (by inspection of *TVP*).

NOTE - This Key Agreement Mechanism has the following properties:

1. Number of passes: 1.

2. Key authentication: this mechanism provides explicit key authentication from *A* to *B* and implicit key authentication from *B* to *A*.

3. Key confirmation: this mechanism provides key confirmation from *A* to *B*.

4. This is a key agreement mechanism since the established key is a one-way function of a random value *r* supplied by *A* and *B*'s private key agreement key. However, since entity *A* may know entity *B*'s public key prior to choosing the value *r*, entity *A* may select approximately *s* bits of the established key, at the cost of generating 2^s candidate values for *r* in the interval between discovering *B*'s public key and sending KT_{A1}.

5. *TVP*: provides entity authentication of *A* to *B* and prevents replay of the key token.

6. Example: an example of this key agreement mechanism (known as Nyberg-Rueppel key agreement) is described in clause B.4.

7. Public key certificates: if *Text1* is used to transfer *A*'s public key certificate, then requirement 2 at the beginning of this clause can be relaxed to the requirement that *B* is in possession of an authenticated copy of the CA's public verification key.

6.4 Key agreement mechanism 4

This key agreement mechanism establishes in two passes a shared secret key between entities *A* and *B* with joint key control without prior exchange of keying information. This mechanism provides neither entity authentication nor key authentication.

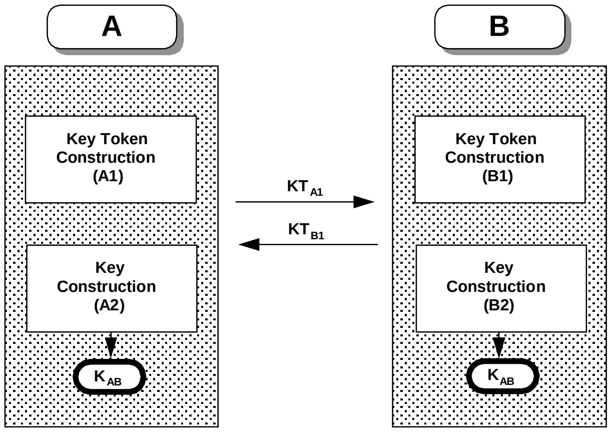

Figure 4 - Key Agreement Mechanism 4

Key Token Construction (A1) *A* randomly and secretly generates r_A in *H*, computes $F(r_A,g)$, constructs the key token

$$KT_{A1} = F(r_A,g) || Text1$$

and sends it to *B*.

Key Token Construction (B1) *B* randomly and secretly generates r_B in *H*, computes $F(r_B,g)$, constructs the key token

$$KT_{B1} = F(r_B,g) || Text2$$

and sends it to *A*.

Key Construction (A2) *A* extracts $F(r_B,g)$ from the received key token KT_{B1} and computes the shared secret key

$$K_{AB} = F(r_A,F(r_B,g))$$

Key Construction (B2) *B* extracts $F(r_A,g)$ from the received key token KT_{A1} and computes the shared secret key

$$K_{AB} = F(r_B,F(r_A,g))$$

NOTE - This Key Agreement Mechanism has the following properties:

1. Number of passes: 2.

2. Key authentication: this mechanism does not provide key authentication. However, this mechanism may be useful in environments where the authenticity of the key tokens is verified using other means. For instance, a hash-code of the key tokens may be exchanged between the entities using a second communication channel. See also Public Key

Transport Mechanism 2. Key confirmation: this mechanism provides no key confirmation.

3. This is a key agreement mechanism since the established key is a one-way function of random values r_A and r_B supplied by *A* and *B* respectively. However, since entity *B* may know $F(r_A, g)$ prior to choosing the value r_B, entity *B* may select approximately *s* bits of the established key, at the cost of generating 2^s candidate values for r_B in the interval between receiving KT_{A1} and sending KT_{B1}.

4. Example: an example of this mechanism (known as Diffie-Hellman key agreement) is described in clause B.5.

6.5 Key agreement mechanism 5

This key agreement mechanism establishes in two passes a shared secret key between entities *A* and *B* with mutual implicit key authentication and joint key control. The following requirements shall be satisfied:

1. Each entity *X* has a private key agreement key h_X in *H* and a public key agreement key $p_X = F(h_X,g)$.
2. Each entity has access to an authenticated copy of the public key agreement key of the other entity. This may be achieved using the mechanisms of clause 8.
3. Both entities have agreed on a common one-way function *w*.

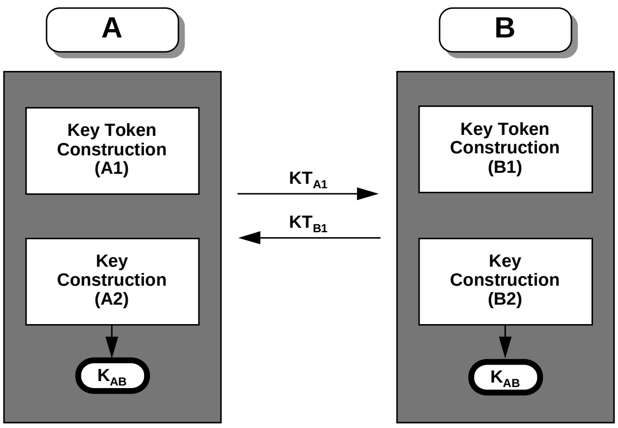

Figure 5 - Key Agreement Mechanism 5

Key Token Construction (A1) *A* randomly and secretly generates r_A in *H*, computes $F(r_A,g)$ and sends the key token

$$KT_{A1} = F(r_A,g)||Text1$$

to *B*.

Key Token Construction (B1) *B* randomly and secretly generates r_B in *H*, computes $F(r_B,g)$ and sends the key token

$$KT_{B1} = F(r_B,g)||Text2$$

to *A*.

Key Construction (B2) *B* extracts $F(r_A,g)$ from the received key token KT_{A1} and computes the shared secret key as

$$K_{AB} = w(F(h_B,F(r_A,g)), F(r_B,p_A))$$

where *w* is a one-way function.

Key Construction (A2) *A* extracts $F(r_B,g)$ from the received key token KT_{B1} and computes the shared secret key as

$$K_{AB} = w(F(r_A,p_B), F(h_A,F(r_B,g)))$$

NOTE - This Key Agreement Mechanism has the following properties:

1. Number of passes: 2.

2. Key authentication: this mechanism provides mutual implicit key authentication. If the data field *Text2* contains a cryptographic check value (on known data) computed using the key K_{AB}, then this mechanism provides explicit key authentication from *B* to *A*.

3. Key confirmation: if the data field *Text2* contains a cryptographic check value (on known data) computed using the key K_{AB}, then this mechanism provides key confirmation from *B* to *A*.

4. This is a key agreement mechanism since the established key is a one-way function of random values r_A and r_B supplied by *A* and *B* respectively. However, since entity *B* may know $F(r_A, g)$ prior to choosing the value r_B, entity *B* may select approximately *s* bits of the established key, at the cost of generating 2^s candidate values for r_B in the interval between receiving KT_{A1} and sending KT_{B1}.

5. Example: An example of this key agreement mechanism (known as the Matsumoto-Takashima-Imai A(0) key agreement scheme) is described in