INTERNATIONAL STANDARD

ISO/IEC 11770-5

Second edition
2020-11

Information security — Key management —

Part 5: Group key management

Sécurité de l'information — Gestion de clés —

Partie 5: Gestion de clés de groupe

Reference number
ISO/IEC 11770-5:2020(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.

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 http://patents.iec.ch).

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*, Subcommittee SC 27, *Information security, cybersecurity and privacy protection*.

This second edition cancels and replaces the first edition (ISO/IEC 11770-5:2011) which has been technically revised.

The main changes compared to the previous edition are as follows:

— the document has been modified to be consistent with use of the key deriviation specifications from ISO/IEC 11770-6;

— the use of a "trapdoor" in key derivation has been removed. Consequently, unlimited forward key chains can no longer be calculated.

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.

Introduction

In some applications, it is necessary for a secret cryptographic key to be shared by a group of entities. Moreover, in some cases the exact membership of a group of entities that share a key may change over time.

This document is concerned with techniques that enable a secret key to be shared by all members of a defined group with the assistance of a trusted third party known as a key distribution centre. Provisions for adding and removing members of a group are also made.

Information security — Key management —

Part 5: Group key management

1 Scope

This document specifies mechanisms to establish shared symmetric keys between groups of entities. It defines:

— symmetric key-based key establishment mechanisms for multiple entities with a key distribution centre (KDC); and

— symmetric key establishment mechanisms based on a general tree-based logical key structure with both individual rekeying and batch rekeying.

It also defines key establishment mechanisms based on a key chain with group forward secrecy, group backward secrecy or both group forward and backward secrecy.

This document also describes the required content of messages which carry keying material or are necessary to set up the conditions under which the keying material can be established.

This document does not specify information that has no relation with key establishment mechanisms, nor does it specify other messages such as error messages. The explicit format of messages is not within the scope of this document.

This document does not specify the means to be used to establish the initial secret keys required to be shared between each entity and the KDC, nor key lifecycle management. This document also does not explicitly address the issue of interdomain key management.

2 Normative references

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

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

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

3 Terms and definitions

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

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

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

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

3.1
active
state of an entity in which the entity can obtain the *shared secret key* (3.24)

3.2
ancestor key
ancestor key of an entity *x*
cryptographic key in a *logical key hierarchy* (3.17) that is assigned to a node on the direct path from the *leaf node* (3.16) corresponding to the *individual key* (3.11) for *x* and the *root node* (3.23)

Note 1 to entry: An ancestor key is either the shared secret key or a key encryption key.

3.3
backward secrecy with interval *T*
security condition in which an entity joining a set of entities at time $t = t_0$ cannot obtain any secret keys established between these entities at any time prior to $t_0 - T$

3.4
batch rekeying with interval *T*
rekeying method in which the *shared secret key* (3.24) and, optionally, *key encryption keys* (3.15) are updated at every time interval *T* (see Clause 4)

3.5
child key
child key for a node *w*
cryptographic key in a *logical key hierarchy* (3.17) assigned to a non-root node *w*

Note 1 to entry: A child key shall be a key encryption key or individual key.

3.6
child node
child node of a node *w*
node in a *tree* (3.25) that is adjacent to *w* and for which *w* lies on the unique path between it and the *root node* (3.23)

3.7
***d*-ary tree**
tree (3.25) where each node has *d child nodes* (3.6) except the *leaf nodes* (3.16) in the tree

3.8
forward secrecy with interval *T*
security condition in which an entity leaving a set of entities at time $t = t_0$ cannot obtain any secret keys established between these entities at any time subsequent to $t_0 + T$

3.9
group backward secrecy
security condition in which an entity joining a set of entities cannot obtain any secret keys previously established between these entities

3.10
group forward secrecy
security condition in which an entity leaving a set of entities cannot obtain any secret keys subsequently established between these entities

3.11
individual key
key shared between the *key distribution centre* (3.14) and each entity

3.12
individual rekeying
rekeying method in which the *shared secret key* (3.24) and, optionally, *key encryption keys* (3.15) are updated when an entity joins or leaves

3.13
key chain
set of cryptographic keys which are not necessarily independent

3.14
key distribution centre
KDC
entity trusted to generate or acquire and distribute keys to entities

3.15
key encryption key
cryptographic key that is used for the encryption or decryption of other keys

[SOURCE: ISO/IEC 19790:2012, 3.62]

3.16
leaf node
node in a *tree* (3.25) that has no *child nodes* (3.6)

3.17
logical key hierarchy
tree (3.25) used for managing the shared secret key and *key encryption keys* (3.15)

3.18
logical key structure
logical structure to manage keys

Note 1 to entry: The choice of the logical key hierarchy is independent of the network topology.

3.19
one-way function
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

[SOURCE: ISO/IEC 11770-3:2015, 3.30]

3.20
one-step key derivation function
OKDF
key derivation function which operates in a single stage, in contrast to key derivation functions involving separate key-extraction and key-expansion stages

[SOURCE: ISO/IEC 11770-6:2016, 3.9]

3.21
random number
time variant parameter whose value is unpredictable

[SOURCE: ISO/IEC 11770-1:2010, 2.39]

3.22
rekeying
process of updating and redistributing the *shared secret key* (3.24) and, optionally, *key encryption keys* (3.15)

Note 1 to entry: This process is executed by the key distribution centre.

3.23
root node
unique identified special node in a *tree* (3.25)

3.24
shared secret key
key which is shared with all the active entities via a key establishment mechanism for multiple entities

3.25
tree
connected, acyclic graph with an identified special node, the *root node* (3.23)

4 Symbols and abbreviated terms

COM(X,Y)	function which generates from the data items X and Y a key designed to be applied as a key for the encryption algorithm in use
CUT(k,S)	function which outputs a substring of length k equal to the least significant bits of a string of bits S
d	number of child nodes for a non-leaf node (see term d-ary tree)
$e(K,Z)$	result of encrypting data Z with a symmetric encryption algorithm using the secret key K
h	number of nodes in the direct path from a leaf node to the root node
$K_{\mathrm{A},i}(x)$	ancestor key for entity x at the i-th layer from the root node
$K_{\mathrm{BW},i}$	backward key for the time instance i
$K_{\mathrm{C},w}$	child key assigned to the node w
$K_{\mathrm{FW},i}$	forward key for the time instance i
K_{I}	individual key
$K_{\mathrm{I}}(x)$	individual key shared between entity x and the key distribution centre
$K_{\mathrm{KE},w}$	key encryption key assigned to a node w
K_{SS}	shared secret key
KDC	key distribution centre
m	number of entities connected to the hub in a star structure
OKDF1	one-step key derivation function that takes a single input as defined in ISO/IEC 11770-6
OKDF6	one-step key derivation function that takes a key and input data as defined in ISO/IEC 11770-6
OWF	one-way function used in the calculation of a key chain
$r_{\mathrm{BW,init}}$	random number to initialize the backward key chain
$r_{\mathrm{FW,init}}$	random number to initialize the forward key chain
T	length of the time interval used in batch rekeying
$\|\|$	binary operator indicating the concatenation of data items

5 Requirements

The key establishment mechanisms specified in this document enable the establishment of shared secret keys within a defined group of entities using multicast communication. In order to maintain security, the mechanisms incorporate a key updating process to be used when a new entity joins or an existing entity leaves the group.

a) The mechanisms specified in this document provide either group backward secrecy and group forward secrecy, or backward and forward secrecy with intervals. The type of group backward/forward secrecy should be chosen depending on the security requirements of the particular application. The type of group backward/forward security property is determined by the choice of rekeying method: individual rekeying provides group backward/forward secrecy, and batch rekeying provides backward/forward secrecy with intervals. The use of batch rekeying requires the choice of a time interval parameter *T*. The rekeying method and parameter setting have a strong influence on the security requirements. Thus, they shall be determined according to the security policy of the application.

b) Symmetric encryption techniques, as required for the mechanisms specified in Clause 6, shall be chosen from amongst those standardized in ISO/IEC 19772.

c) The shared secret key is established using either a secure or an insecure communication channel. Each individual key shall be exchanged between the KDC and each entity using a secure channel in order to allow secure communication. A secure communication channel is one where an attacker cannot eavesdrop or tamper with messages in the channel.

d) The key establishment mechanisms in this document require the use of random numbers to generate the shared secret key, and optionally, key encryption keys. For means of generating random numbers, see ISO/IEC 18031.

e) Annex A defines object identifiers in accordance with ISO/IEC 9834 (all parts) that shall be used to identify the mechanisms specified in this document. Any change to the specification of the mechanisms resulting in a change of functional behaviour results in a change of the object identifier assigned to the mechanisms.

6 Tree-based key establishment mechanisms

6.1 General model

Use of the mechanisms specified in this document enables the establishment of a secret key shared by all the entities in a defined group. This enables any member of the group to send an encrypted message to all the other group members such that only group members (and the key distribution centre) can decrypt it. The mechanisms also enable the key distribution centre to update the established secret key to ensure that an encrypted message can only be decrypted by entities who are group members at that time the message was encrypted.

Figure 1 shows the general model of key establishment for multiple entities, in which the key distribution centre can communicate with all the entities. The communication between the key distribution centre and entities does not need to be secure. The key distribution centre and each entity shall share a distinct individual key. The key distribution centre is responsible for distributing the shared secret key to all the active entities. A join/leave request is shown as (1) and the distribution of keys to the entities as (2), (3), ..., (n + 1). From (2) onward, the order in which the updates take place is not important.

NOTE If one of the entities that knows the shared secret key cannot be contacted for a period of time, that entity can miss a key update message, and as a result will not be able to compute the updated shared secret key.

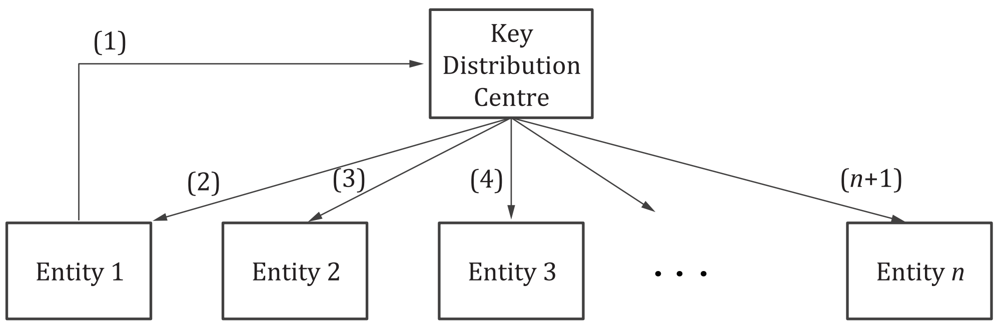

Figure 1 — General model of key establishment for multiple entities

6.2 Joining process

An entity sends a join request to the key distribution centre in order to start the process of obtaining the shared secret key. If individual rekeying is in use, as necessary to support group backward/forward secrecy, then the key distribution centre shall execute the rekeying process after the joining request has been accepted. However, if batch rekeying is in use, supporting backward/forward secrecy with intervals, then the rekeying process is not automatically executed at this point.

6.3 Leaving process

An entity sends a leave request to the key distribution centre in order to stop obtaining the shared secret key. If individual rekeying is in use, then the key distribution centre shall execute the rekeying processes after an entity has left. However, if batch rekeying is in use, then the key distribution centre shall record the leaving entities for the next rekeying interval.

NOTE When batch rekeying is in use, the entity leaving the group can still decrypt communications sent within the group until the next batch rekeying takes place.

6.4 Rekeying process

This process involves the key distribution centre updating the secret key shared with the entities in a group; it can also involve updating key encryption keys. If individual rekeying is in use, then this process shall be performed as part of the joining and leaving processes. If batch rekeying is in use, it shall be performed at regular time intervals.

6.5 Logical key structure

6.5.1 General

Key establishment mechanisms can be classified according to the logical structure defined by the means used to distribute the shared secret key from the key distribution centre to the active entities in the group. Three specific logical key structures are defined in 6.5.2 to 6.5.4.

6.5.2 Star-based structure

In a star-based structure, the shared secret key is directly encrypted for distribution using the individual keys assigned to the entities. An example of a star-based structure with six key encryption keys is shown in Figure 2, where the double circle denotes the key distribution centre.

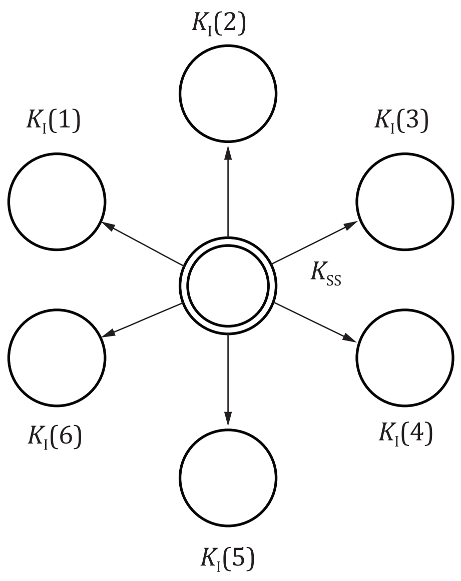

Figure 2 — Star-based structure

6.5.3 *d*-ary tree-based structure

A tree-based structure can reduce the number of key encryption keys held by individual entities. Figure 3 shows the binary tree structure where $d = 2$. A shared secret key is assigned to the root node of the tree. Each individual key is assigned to the leaf nodes of the tree. Additionally, key encryption keys are assigned to the other nodes. The key encryption keys are shared by multiple entities whose individual keys are assigned to the descendant of the node to which the key encryption key is assigned. The communication cost of the leaving process may be reduced by using key encryption keys. Each entity has all the keys assigned to the nodes on the path from the root node to the leaf node, to which the individual key of the entity is assigned. Thus, the number of keys an entity has is proportional to the logarithm of the total number of active entities.

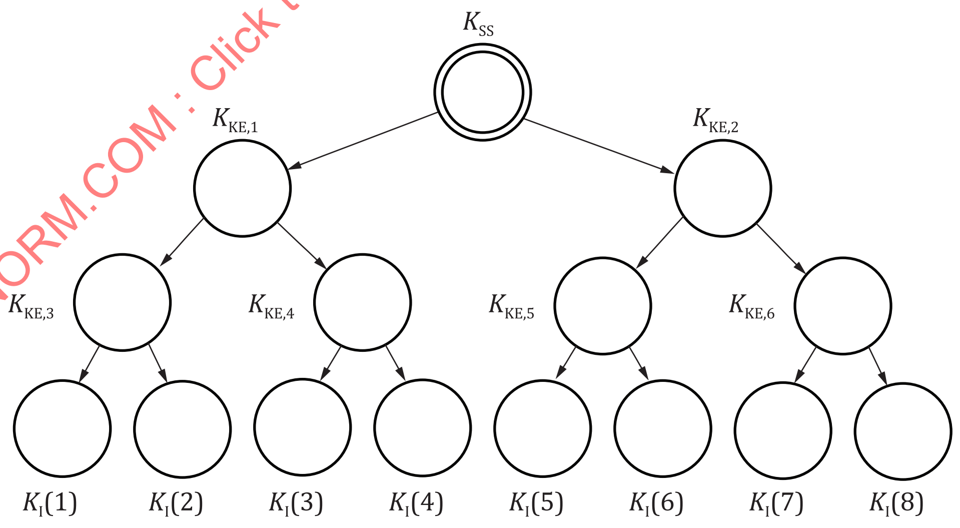

Figure 3 — *d*-ary tree-based structure

6.5.4 General tree-based structure

A general tree-based structure can be used as the logical key structure. The general tree-based structure makes use of a *d*-ary tree-based structure where *m* entities construct a cluster. This structure can be considered as a hybrid of the star-based structure with *m* clients and the *d*-ary tree-based structure.

This structure can be used to optimize the efficiency of key establishment mechanisms (see Annex B). Figure 4 shows the tree-based structure where $d = 2$ and $m = 4$. The general tree-based structure contains a d-ary tree-based structure, however, the opposite does not hold. For example, the tree-based structure in Figure 4 is not a d-ary tree-based structure.

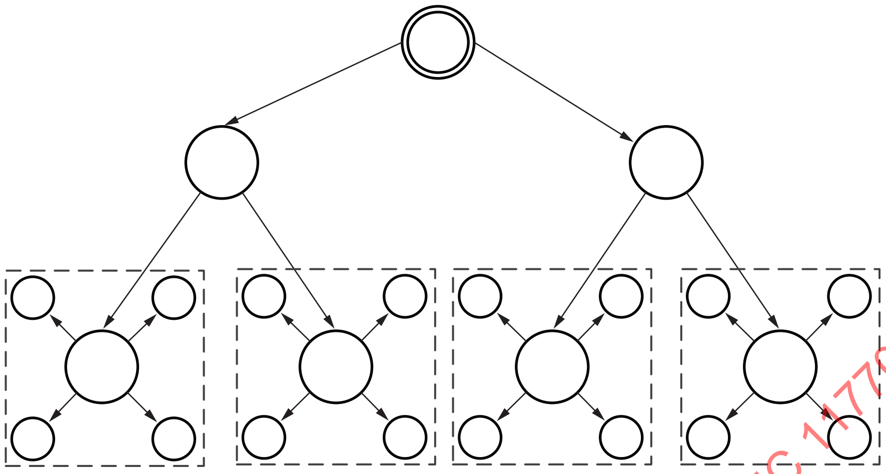

Figure 4 — General tree-based structure

6.6 Symmetric key-based key establishment mechanisms

6.6.1 General

This document defines two symmetric key-based key establishment mechanisms for multiple entities based on a general tree-based structure: 1) a mechanism with individual rekeying and 2) a mechanism with batch rekeying. In the mechanism with individual rekeying, the rekeying process is executed whenever an entity joins or leaves.

6.6.2 Mechanism 1 — Key establishment mechanism with individual rekeying

This mechanism is based on a tree-based structure with individual rekeying.

a) Joining process

It is assumed that there is a set of n active entities $\{u_1, u_2, ..., u_n\}$, and the entity u_{n+1} joins. Let $K_{A,l}(u_i)$ be the ancestor key of entity u_i that is assigned to the l-th layer from the root node of the logical key hierarchy. Let h denote the height of the logical key hierarchy.

1) The entity u_{n+1} sends a join request to the key distribution centre.

2) The key distribution centre assigns the individual key of u_{n+1} (i.e. $K_I(u_{n+1})$) to a leaf node of the logical key hierarchy.

3) The key distribution centre generates random numbers and updates the ancestor keys of the individual key of u_{n+1} using these numbers. K_{SS}, $K_{A,1}(u_{n+1})$, $K_{A,2}(u_{n+1})$, ..., $K_{A,h}(u_{n+1})$ are updated to K'_{SS}, $K'_{A,1}(u_{n+1})$, $K'_{A,2}(u_{n+1})$, ..., $K'_{A,h}(u_{n+1})$, respectively.

4) The key distribution centre encrypts each updated key with the old key, and broadcasts it. That is, $e(K_{SS}, K'_{SS})$, $e(K_{A,1}(u_{n+1}), K'_{A,1}(u_{n+1}))$, $e(K_{A,2}(u_{n+1}), K'_{A,2}(u_{n+1}))$, ..., and $e(K_{A,h}(u_{n+1}), K'_{A,h}(u_{n+1}))$ are broadcast.

5) Each entity obtains the updated keys using the old keys.

6) The key distribution centre encrypts the updated keys $K'_{\mathrm{SS}}||\ K'_{\mathrm{A},1}(u_{n+1})||\ K'_{\mathrm{A},2}(u_{n+1})||...||\ K'_{\mathrm{A},h}(u_{n+1})$ by the individual key of u_{n+1}, and sends $e(K_{\mathrm{I}}(u_{n+1}), K'_{\mathrm{SS}}||\ K'_{\mathrm{A},1}(u_{n+1})||\ K'_{\mathrm{A},2}(u_{n+1})||...||\ K'_{\mathrm{A},h}(u_{n+1}))$ to u_{n+1}.

7) The entity u_{n+1} obtains the keys.

b) Leaving process

It is assumed that there are n active entities $\{u_1, u_2, ..., u_n\}$, and the entity u_j $(1 \leq j \leq n)$ leaves.

1) The key distribution centre generates random numbers and updates the ancestor keys of the individual key of u_j using these numbers. K_{SS}, $K_{\mathrm{A},1}(u_j)$, $K_{\mathrm{A},2}(u_j)$, ..., $K_{\mathrm{A},h}(u_j)$ are updated to K'_{SS}, $K'_{\mathrm{A},1}(u_j)$, $K'_{\mathrm{A},2}(u_j)$, ..., $K'_{\mathrm{A},h}(u_j)$, respectively.

2) The key distribution centre encrypts each updated key with all the child keys except the individual key of u_j and broadcasts them. For example, the K'_{SS} is encrypted with the child keys $K_{\mathrm{C},1}$, $K_{\mathrm{C},2}$, ..., $K_{\mathrm{C},d}$, and $e(K_{\mathrm{C},1}, K'_{\mathrm{SS}})$, $e(K_{\mathrm{C},2}, K'_{\mathrm{SS}})$, ..., and $e(K_{\mathrm{C},d}, K'_{\mathrm{SS}})$ are broadcast.

NOTE 1 In the case that child keys have been updated, the updated child keys are used.

3) Each entity obtains the updated keys using the child keys.

Usage example 1

This example demonstrates the joining process of Mechanism 1 in the scenario illustrated in Figure 5. It is assumed that the key distribution centre uses the logical key hierarchy in Figure 6 and the entity H is joining. Recall that each active entity has all the keys assigned to the nodes on the path of the logical key hierarchy from the leaf node corresponding to the individual key of the entity to the root node. For example, entity A has $K_{\mathrm{I}}(A)$, $K_{\mathrm{A},3}$, $K_{\mathrm{A},1}$, and K_{SS}.

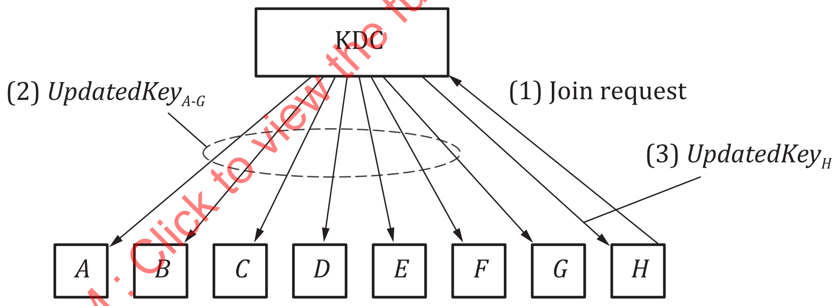

Figure 5 — Joining process of Mechanism 1 — Mechanism with individual rekeying

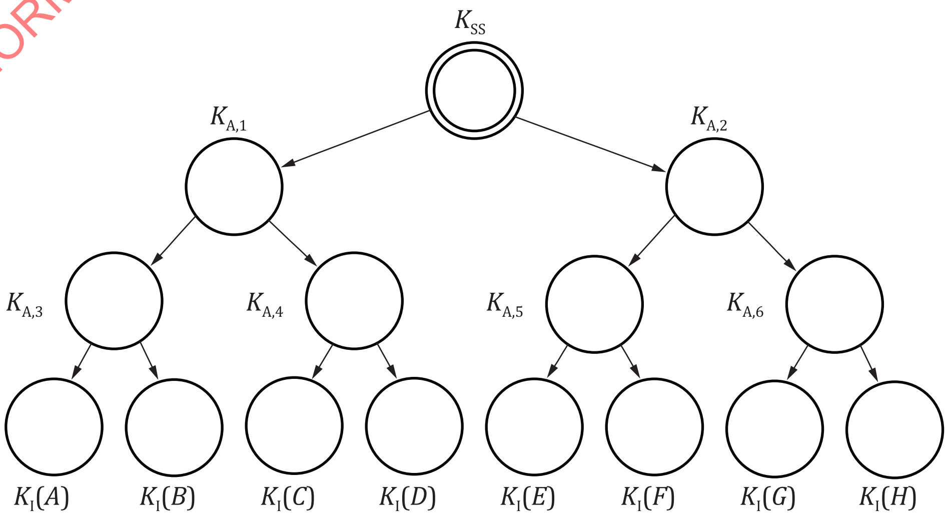

Figure 6 — Logical key procedure

The updated keys ($UpdatedKey_{A\text{-}G}$), broadcast by the key distribution centre to A, B, C, D, E, F, and G is:

$$UpdatedKey_{A\text{-}G} = e(K_{SS}, K'_{SS}) || e(K_{A,2}, K'_{A,2}) || e(K_{A,6}, K'_{A,6})$$

The updated keys ($UpdatedKey_H$), sent by the key distribution centre to H is:

$$UpdatedKey_H = e(K_I(H), K'_{SS} || K'_{A,2} || K'_{A,6})$$

1) H sends a join request to the key distribution centre.

2) The key distribution centre generates and broadcasts $UpdatedKey_{A\text{-}G}$ *to* A, B, C, D, E, F, and G.

3) The key distribution centre generates and sends $UpdatedKey_H$ to H.

NOTE 2 The key distribution centre and H share the individual key of H in advance.

Usage example 2

This example demonstrates the leaving process of Mechanism 1 in the scenario illustrated in Figure 7. It is assumed that the key distribution centre uses the logical key hierarchy in Figure 6 and the entity H is leaving.

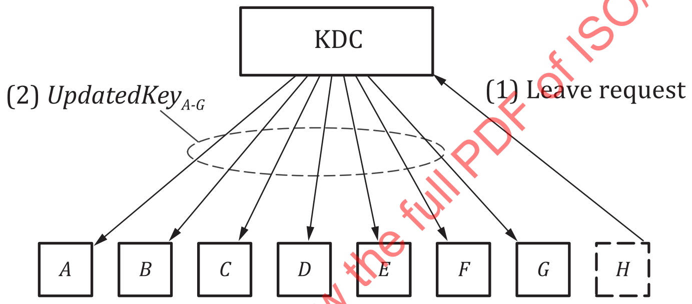

Figure 7 — Leaving process of Mechanism 1 – Mechanism with individual rekeying

The form of the updated keys ($UpdatedKey_{A\text{-}G}$), broadcast by the key distribution centre to A, B, C, D, E, F, and G is:

$$UpdatedKey_{A\text{-}G} = e(K_{A,1}, K'_{SS}) || e(K'_{A,2}, K'_{SS}) || e(K_{A,5}, K'_{A,2}) || e(K'_{A,6}, K'_{A,2}) || e(K_I(G), K'_{A,6})$$

1) H sends a leave request to the key distribution centre.

2) The key distribution centre generates and broadcasts $UpdatedKey_{A\text{-}G}$ to A, B, C, D, E, F, and G.

6.6.3 Mechanism 2 — Key establishment mechanism with batch rekeying

In the mechanism with batch rekeying with interval T, the rekeying process is periodically executed every time interval T.

a) Joining process

It is assumed that there are n active entities $\{u_1, u_2, ..., u_n\}$, and the entity u_{n+1} joins.

1) The entity u_{n+1} sends a join request to the key distribution centre.

2) The key distribution centre assigns the individual key of u_{n+1} (i.e. $K_I(u_{n+1})$) to a leaf node of the logical key hierarchy.

3) The key distribution centre encrypts the ancestor keys of the individual key of u_{n+1} by the individual key of u_{n+1}. Then, the key distribution centre sends $e(K_I(u_{n+1}), K_{SS}||K_{A,1}(u_{n+1})||K_{A,2}(u_{n+1})|| \ldots ||K_{A,h}(u_{n+1}))$ to u_{n+1}.

4) The entity u_{n+1} obtains K_{SS} and $K_{A,1}(u_{n+1})$, $K_{A,2}(u_{n+1})$, ..., $K_{A,h}(u_{n+1})$.

b) Leaving process

The leaving entity sends a leave request to the key distribution centre.

NOTE 1 Rekeying is not executed in the leaving process of the mechanism with batch rekeying.

c) Rekeying process

This process executed at a regular time interval T. It is assumed that there is a set of n active entities $\{u_1, u_2, \ldots, u_n\}$ and that the entities of the set $\{u_{i,1}, u_{i,2}, \ldots, u_{i,k}\}$ left during a rekeying interval. k is the number of leaving entities during a rekeying interval i.

1) The key distribution centre generates random numbers and updates the ancestor keys of the individual keys for $u_{i,1}, u_{i,2}, \ldots, u_{i,k}$ using the random numbers. K_{SS}, $K_{A,1}(u_{i,1})$, $K_{A,1}(u_{i,2})$, ..., $K_{A,1}(u_{i,k})$, $K_{A,2}(u_{i,1})$, $K_{A,2}(u_{i,2})$, ..., $K_{A,2}(u_{i,k})$, ..., $K_{A,h}(u_{i,1})$, $K_{A,h}(u_{i,2})$, ..., $K_{A,h}(u_{i,k})$ are updated to K'_{SS}, $K'_{A,1}(u_{i,1})$, $K'_{A,1}(u_{i,2})$, ..., $K'_{A,1}(u_{i,k})$, $K'_{A,2}(u_{i,1})$, $K'_{A,2}(u_{i,2})$, ..., $K'_{A,2}(u_{i,k})$, ..., $K'_{A,h}(u_{i,1})$, $K'_h(u_{i,2})$, ..., $K'_{A,h}(u_{i,k})$, respectively.

2) The key distribution centre encrypts each updated key with all the child keys except the individual keys of $u_{i,1}, u_{i,2}, \ldots, u_{i,k}$ and broadcasts them. For example, the K'_{SS} is encrypted with the child keys $K_{C,1}, K_{C,2}, \ldots, K_{C,d}$, and $e(K_{C,1}, K'_{SS})$, $e(K_{C,2}, K'_{SS})$, ..., and $e(K_{C,d}, K'_{SS})$ are broadcast.

NOTE 2 In the case that child keys have been updated, the updated child keys are used.

3) Each entity obtains the updated keys using the child keys.

Usage example 3

This example demonstrates the joining process of Mechanism 2 in the scenario illustrated in Figure 8. It is assumed that the key distribution centre uses the logical key hierarchy in Figure 6 and the entity H is joining. Recall that each entity has all the keys assigned to the nodes on the path of the logical key hierarchy from the leaf node corresponding to the individual key of the entity to the root node. For example, entity A has $K_I(A)$, $K_{A,3}$, $K_{A,1}$, and K_{SS}.

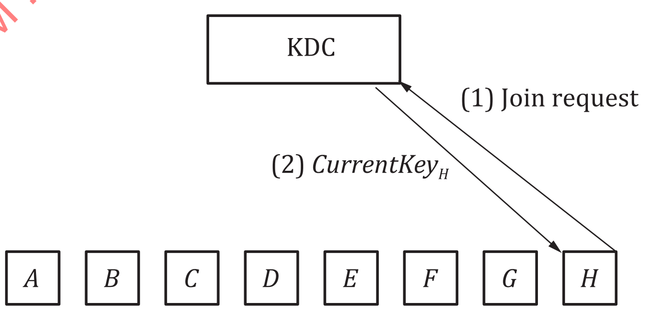

Figure 8 — Joining process of Mechanism 2 – Mechanism with batch rekeying

The form of the updated keys ($CurrentKey_H$), sent by the key distribution centre to H is:

$$CurrentKey_H = e(K_I(H), K_{SS}||K_{A,2}||K_{A,6})$$

1) H sends a join request to the key distribution centre.

2) The key distribution centre generates and sends $CurrentKey_H$ to H.

Usage example 4

This example demonstrates the leaving process of Mechanism 2 in the scenario illustrated in Figure 9. It is assumed that the key distribution centre uses the logical key hierarchy in Figure 6 and the entities *A* and *H* have left.

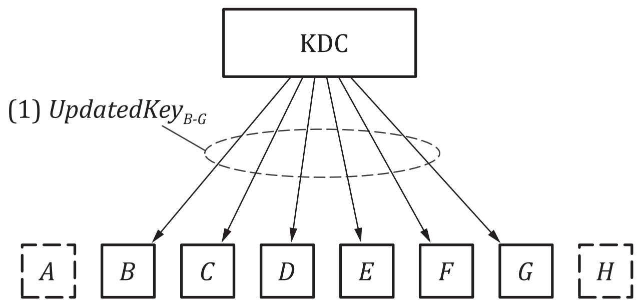

Figure 9 — Rekeying process of Mechanism 2 – Mechanism with batch rekeying

The updated keys ($UpdatedKey_{B\text{-}G}$), broadcast by the key distribution centre to *B*, *C*, *D*, *E*, *F*, and *G* is:

$$UpdatedKey_{B\text{-}G} = e(K'_{A,1},K'_{SS})||e(K'_{A,2},K'_{SS})||e(K'_{A,3},K'_{A,1})||e(K_{A,4},K'_{A,1})||$$

$$e(K_{A,5},K'_{A,2})||e(K'_{A,6},K'_{A,2})||e(K_I(B),K'_{A,3})||e(K_I(G),K'_{A,6})$$

The key distribution centre generates and broadcasts $UpdatedKey_{B\text{-}G}$ to *B*, *C*, *D*, *E*, *F*, and *G*.

7 Key chain-based group key management with limited forward key chain

7.1 General model

In order to limit the validity period of keys within a static group membership, key chains are useful. Key chains can limit access to encrypted information in future sessions, past sessions or both. In a key chain the individual keys are dependent on each other. Using a start value, a first key is calculated. The second key is calculated to be dependent on the first one, the third key calculated to be dependent on the second key, etc. If the decentralized entity is able to perform the calculation, it allows for efficient decentralized key generation. Key chain length is limited to a predetermined number of keys. Any particular instance of a key in the key chain, and the time interval for which it is valid, determines the time interval in which the encrypted information can be decrypted using that particular key.

In this document, two types of key chains are considered. A forward key chain limits access to encrypted information in the future (i.e. group forward secrecy), whereas a backward key chain limits access to encrypted information in the past (i.e. group backward secrecy). Because the number of keys in a forward and backward key chain is finite, the number of keys in these chains shall be carefully chosen at system implementation.

The calculation of keys within the chains is based on one-way functions. One-way functions allow calculation in one direction only. This means that starting with the result of a one-way function, calculation of the start value cannot be performed. A key derivation function from ISO/IEC 11770-6 shall be selected for the one-way function.

Special measures shall be implemented to prevent collusion attacks, in which one or more participants knowingly provide information to an attacker. If one entity has access to one part of a key chain, and another entity has access to another part of the same key chain, they can reconstruct the key chain from the lowest start of both parts of the key chain until the highest end of both parts of the key chain.

The keys of a key chain shall be handled like secret keys, whose forwarding to other entities needs to be prevented.

7.2 Calculations by the key distribution centre

7.2.1 Key chains

The key distribution entity needs to set up the key chains prior to normal operation. This means that, depending on what is needed, the forward and/or backward key chain shall be defined. The result for the forward chain would then look like:

$$K_{\mathrm{FW}} = \{K_{\mathrm{FW},0}, K_{\mathrm{FW},1}, \ldots, K_{\mathrm{FW},n\text{-}1}, K_{\mathrm{FW},n}\}$$

and for the backward chain;

$$K_{\mathrm{BW}} = \{K_{\mathrm{BW},0}, K_{\mathrm{BW},1}, \ldots, K_{\mathrm{BW},T\text{-}1}, K_{\mathrm{BW},T}, \ldots\}$$

Both key chains are calculated using one-way functions.

In this case, the start values are $K_{\mathrm{FW},n}$ and $K_{\mathrm{BW},0}$. The index n of the start value $K_{\mathrm{FW},n}$ shall be chosen appropriately based on the application and implementation aspects, because it means the end of the forward key chain.

7.2.2 Group forward secrecy

The key distribution centre constructs the forward key chain using the given start value $K_{\mathrm{FW},n}$. Any of the keys contained in the limited chain can be evaluated by:

$$K_{\mathrm{FW},n\text{-}i\text{-}1} = \mathrm{OWF}(K_{\mathrm{FW},n\text{-}i}),\ i = 0,1,2,\ldots,n\text{-}1$$

where OWF is a one-way function. The iterative relation can be written as:

$$K_{\mathrm{FW},n-T} = \mathrm{OWF}^{T}\left(K_{\mathrm{FW},n}\right) := \underbrace{\mathrm{OWF}\left(\mathrm{OWF}\left(\ldots\mathrm{OWF}\left(K_{\mathrm{FW},n}\right)\right)\right)}_{T\ \text{times}},\ T = 1,2,3,\ldots,n$$

Figure 10 also outlines the calculation of the forward key chain $K_{\mathrm{FW},n}$ ... $K_{\mathrm{FW},0}$.

This chain achieves forward secrecy with interval T since knowledge of $K_{\mathrm{FW},T}$ does not allow calculation of any key $K_{\mathrm{FW},t}$ for $t > T$.

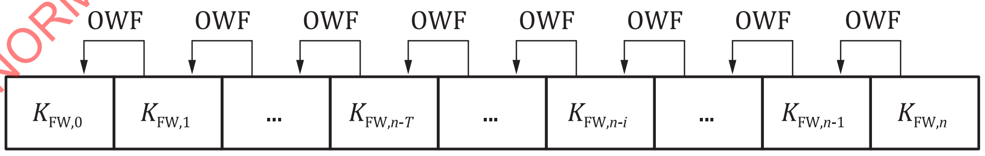

Figure 10 — Construction of a forward key chain by use of a one-way function

Usage example 5

In this example the forward key chain construction of the key distribution centre uses the function OKDF1 from ISO/IEC 11770-6 as a one-way function and performs the following steps.

1) The key distribution centre generates a secure random number as a start value $r_{\mathrm{FW,init}}$.

2) The key distribution centre uses OKDF1 for hashing the random start value to a fixed size of the appropriate length. The output is used as the first key $K_{\mathrm{FW},n}$. That is, $K_{\mathrm{FW},n} = \mathrm{OKDF1}(r_{\mathrm{FW,init}})$ is calculated.

3) The next element of the chain is constructed using $K_{\mathrm{FW},n}$ as input for OKDF1 and using the output of the function as $K_{\mathrm{FW},n\text{-}1}$: $K_{\mathrm{FW},n\text{-}1}$ = OKDF1($K_{\mathrm{FW},n}$).

4) New keys of the chain are generated using the approved key derivation method. The input is always the current last key of the chain $K_{\mathrm{FW},n\text{-}i\text{+}1}$ and the output is $K_{\mathrm{FW},n\text{-}\mathrm{i}}$: $K_{\mathrm{FW},n\text{-}\mathrm{i}}$ = OKDF1($K_{\mathrm{FW},n\text{-}i\text{+}1}$).

5) The calculation finishes as soon as the key distribution centre declares one final value $K_{\mathrm{FW},0}$.

7.2.3 Group backward secrecy

Group backward secrecy can be realized by defining a backward key chain. Any of the keys contained in the chain can be evaluated by:

$$K_{\mathrm{BW},i+1} = \mathrm{OWF}(K_{\mathrm{BW},i}),\ i = 0,1,2,\ldots$$

where OWF is a one-way function. The iterative relation can be written as:

$$K_{\mathrm{BW},T} = \mathrm{OWF}^{T}\left(K_{\mathrm{BW},0}\right) := \underbrace{\mathrm{OWF}\left(\mathrm{OWF}\left(\ldots\mathrm{OWF}\left(K_{\mathrm{BW},0}\right)\right)\right)}_{T\text{ times}},\ T=1,2,3,\ldots$$

The key distribution centre is able to continuously calculate the backward key chain $K_{\mathrm{BW},0}$ ... $K_{\mathrm{BW},T}$ ... as it is outlined in Figure 11.

This chain achieves backward secrecy with interval T since it does not allow access to encrypted information for $t < T$ in the case that the user entity knows $K_{\mathrm{BW},T}$. Any key chain $K_{\mathrm{BW},i}$ ... $K_{\mathrm{BW},T}$,... gives restriction for any time $t < t_i$.

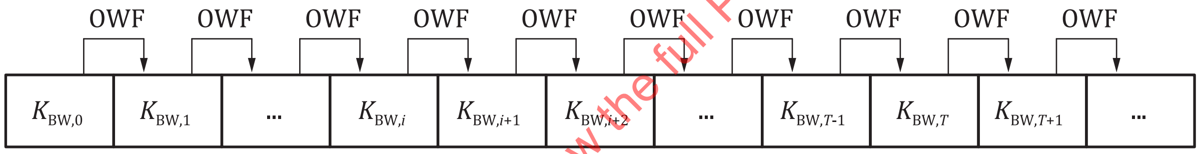

Figure 11 — Construction of a backward key chain using a one-way function

Usage example 6

In this example, the backward key chain construction of the key distribution centre uses the key derivation function OKDF1 from ISO/IEC 11770-6 as a one-way function and performs the following steps.

1) The key distribution centre generates a secure random number $r_{\mathrm{BW,init}}$ as a start value.

2) The key distribution centre uses OKDF1 for hashing the random start value to a fixed size as required. The output is used as the first key $K_{\mathrm{BW},0}$. That is, $K_{\mathrm{BW},0}$ = OKDF1($r_{\mathrm{BW,init}}$) is calculated.

3) The next element of the chain is constructed using $K_{\mathrm{BW},0}$ as input for OKDF1 and using the output of the function as $K_{\mathrm{BW},1}$: $K_{\mathrm{BW},1}$ = OKDF1($K_{\mathrm{BW},0}$).

4) New keys of the chain are generated using OKDF1. The input is always the current last key of the chain $K_{\mathrm{BW},i}$ and the output is $K_{\mathrm{BW},i+1}$: $K_{\mathrm{BW},i+1}$ = OKDF1($K_{\mathrm{BW},i}$).

7.2.4 Forward and backward secrecy

For combined forward and backward secrecy, the individual key chains are connected to allow restricted access in both directions. This is realized by another function COM used to combine the