Journal of Southeast University (English Edition) Vol. 22, No. 1, pp. 54 – 58 Mar. 2006 ISSN 1003—7985

A new group key management scheme based on keys tree, XOR operation and one-way function

Zhang Yong Zhang Yi Wang Weinong

(Network and Information Center, Shanghai Jiaotong University, Shanghai 200030, China)

Abstract: By introducing XOR operation and one-way function chains to group key management schemes based on the keys tree, a new group key management scheme based on the keys tree, XOR operation and one-way function chains is proposed. Initialization, member adding and member evicting operations are introduced. The new scheme is compared with three other group key management schemes which are based on the keys tree: SKDC, LKH, and OFT. As far as transmission, computation and storage costs are concerned, the performance of the new group key management scheme is the best. The security problem of the new scheme is analyzed. This new scheme provides backward and forward security, i. e., newly admitted group members cannot read previous multicast messages and evicted members cannot read future multicast messages, even with collusion by many arbitrarily evicted members.

Key words: secure group communication; group key management; keys tree; one-way function

The maturity of Internet multicast techniques results in more and more multicast-based applications, for example, stock information distribution, remote video conferences, remote collaboration systems, pay per view, network games, distributed simulations, etc. Although these applications can be implemented by unicast techniques, multicast can save the network bandwidth resources and the resources of senders. Multicast can also reduce latency.

There are many security problems in multicast communication, for example, confidentiality of multicast data, single source authentication of multicast data and group member access control[1, 2], etc. Until these security problems are solved, a broad application of multicast is impossible. Among these, the most important is confidentiality of multicast data. This is handled by a group key which is shared by all group members.

Most existing group key management schemes can be classified into two classes: The first class is group key agreement schemes which are based on Diffie-Hellman key exchange protocol, for example, GDH[3]. Due to great computational cost and latency, these schemes do not adapt to large dynamic groups. The second class is centralized group key distribution schemes which are based on the keys tree, for example, LKH[4-6], OFT[7]. Because the transmissional cost is $O(\lg n)$ and the storage cost of each group member is $O(\lg n)$ in all these schemes, these kinds of schemes are better choices for large dynamic groups.

In this paper we present a new group key management scheme based on the keys tree, XOR operation and one-way function. We analyze the security of this new scheme in detail. We compare our scheme with other schemes which are based on the keys tree.

1 Group Key Management Scheme Based on Keys Tree, XOR Operation and One-Way Function

Our scheme manages the update of the group key by the keys tree. This is similar to LKH. In our scheme keys tree is a binary tree. It is depicted in Fig. 1.

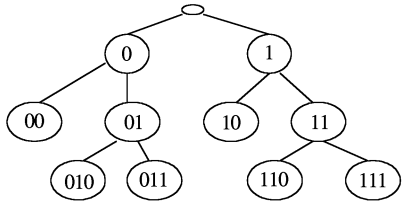

Fig. 1 Keys tree and node number

Except for the root node every node has a number. The generation of the numbers is similar to Huffman coding. It is depicted in Fig. 1. Every leaf node corresponds to a group member. The number of a leaf node is the number of a corresponding group member. Every node has a key. If the number of one node is N, then the key of this node is S_N. The root node corresponds to the group key GK.

If the number of some group member is $N_1N_2\ldots N_m$, then this group member knows the following

Received 2005-04-27.

Biographies: Zhang Yong (1976—), male, graduate; Wang Weinong (corresponding author), male, doctor, professor, wn_wang@sjtu.edu.cn.

keys: $S_{N_1N_2\cdots\bar{N}_m}$, $S_{N_1N_2\cdots\bar{N}_{m-1}}$, …, $S_{\bar{N}_1}$ ($\bar{N}_i$ is the opposite of N_i.) and the group key GK. For example, in Fig. 1 group member 010 knows S_{011}, S_{00}, S_1 and GK.

In the following sections, $\oplus$ represents XOR operation. H is a one-way function. If the output length of a kind of one-way function is equal to the length of keys, it can be used in our scheme. In $H^l(M)$, l is the computation number of H. S'_N is new S_N. GK′ is the new group key.

1.1 Initialization of keys tree

Based on the initial number of group members, the group key manager builds a keys tree which is like the one depicted in Fig. 1. The group key manager sends corresponding keys to each member by the secure unicast tunnels between theses members and the group key manager. Generally, a group member must be authenticated in secure group communications. In the group key distribution scheme, the group key manager is the best party who does the authentication operation. The way to do authentication and build secure unicast tunnels depends on the secure policy of group communication.

When the group key manager sends keys update messages, he must send the digital signature of these messages. Each member can be sure that these messages are from the group key manager. We can do this by deploying a PKI system in group communication applications. In the following sections the corresponding signatures are included in the messages which are sent by the group key manager. For clarity we just pass over it.

1.2 Adding a new group member

If a new member joins the group, the group key manager will select a leaf node whose height is the lowest. Assume that the number of this leaf node is $N_1N_2\cdots N_m$. The number of the new member will be $N_1N_2\cdots N_m1$, while the member whose number is $N_1N_2\cdots N_m$ will change its number to $N_1N_2\cdots N_m0$. The group key manager generates random keys: $S_{N_1N_2\cdots N_m0}$, $S_{N_1N_2\cdots N_m1}$ and ΔS. It computes $H^1(\Delta S)$, …, $H^m(\Delta S)$ and computes S'_{N_1}, $S'_{N_1N_2}$, …, $S'_{N_1N_2\cdots N_m}$ and group key GK′ by Eqs. (1) and (2).

$$S'_{N_1N_2\cdots\bar{N}_n} = S_{N_1N_2\cdots\bar{N}_n} \oplus H^{m-n}(\Delta S) \qquad n = 1, 2, \ldots, m \tag{1}$$

$$\mathrm{GK}' = \mathrm{GK} \oplus H^m(\Delta S) \tag{2}$$

The group key manager generates the following keys tree update message and multicasts it to the group: The number of the new member is $N_1N_2\cdots N_m1$; $H^m(\Delta S)\oplus S_{N_1}$, $H^{m-1}(\Delta S)\oplus S_{N_1N_2}$, …, $H^1(\Delta S)\oplus S_{N_1N_2\cdots N_m}$.

Those members whose numbers begin with $\bar{N}_1$ can compute $H^m(\Delta S)$ by their known S_{N_1} and $H^m(\Delta S)\oplus S_{N_1}$ which is included in the keys tree update message, and can compute the new group key by Eq. (2).

Those members whose numbers begin with $N_1N_2\cdots\bar{N}_n$ ($n = 2, 3, \ldots, m$) can compute $H^{m-n+1}(\Delta S)$ by $S_{N_1N_2\cdots N_n}$ and $H^{m-n+1}(\Delta S)\oplus S_{N_1N_2\cdots N_n}$ which is included in the keys tree update message. Then these members compute $H^{m-n+2}(\Delta S)$, …, $H^m(\Delta S)$ and compute $S'_{N_1N_2\cdots\bar{N}_{n-1}}$, …, $S'_{\bar{N}_1}$ and the new group key by Eqs. (1) and (2).

The group key manager sends $S_{N_1N_2\cdots N_m0}$, $S'_{N_1N_2\cdots\bar{N}_m}$, $S'_{N_1N_2\cdots\bar{N}_{m-1}}$, …, $S'_{\bar{N}_1}$ and the new group key to member $N_1N_2\cdots N_m1$ by their unicast secure tunnels. The group key manager sends $S_{N_1N_2\cdots N_m1}$ and ΔS to member $N_1N_2\cdots N_m0$ by their secure unicast tunnels. Member $N_1N_2\cdots N_m0$ can compute $H^1(\Delta S)$, …, $H^m(\Delta S)$ and compute $S'_{N_1N_2\cdots\bar{N}_m}$, …, $S'_{\bar{N}_1}$ and the new group key.

The changes of the keys tree are depicted in Fig. 2 when member 001 joins the group.

$$S'_{01} = S_{01}\oplus\Delta S, \quad S'_1 = S_1\oplus H^1(\Delta S)$$

$$\mathrm{GK}' = \mathrm{GK}\oplus H^2(\Delta S)$$

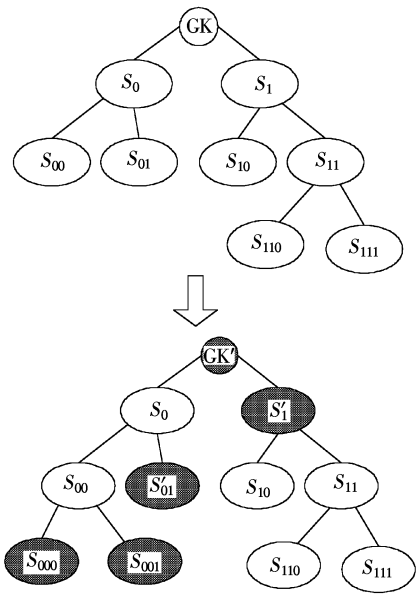

Fig. 2 Changes of keys tree when member 001 joins into the group

The keys tree update message which the group key manager will multicast is: The number of the new member is 001; $H^1(\Delta S)\oplus S_{00}$, $H^2(\Delta S)\oplus S_0$.

For members 10, 110 and 111, they can compute $H^2(\Delta S)$ from known S_0 and $H^2(\Delta S)\oplus S_0$ which is included in the keys tree update message. So these members can compute a new group key GK′.

Member 01 can compute $H^1(\Delta S)$ from known S_{00} and $H^1(\Delta S)\oplus S_{00}$ which is included in the keys tree update message and compute $H^2(\Delta S)$. So mem-

ber 01 can compute S'_1 and the new group key GK′.

Members 000 and 001 can get corresponding keys and a new group key from their secure unicast tunnels with the group key manager.

1.3 Evicting a group member

If a group member whose number is $N_1N_2\ldots N_m$ leaves the group, the group key manager will generate a keys tree update message and multicast it: The number of the group member who leaves the group is $N_1N_2\ldots N_m$; $H^1(S_{N_1N_2\ldots N_m}) \oplus S_{N_1N_2\ldots N_{m-1}}$, $H^2(S_{N_1N_2\ldots N_m}) \oplus S_{N_1N_2\ldots N_{m-2}}$, …, $H^{m-1}(S_{N_1N_2\ldots N_m}) \oplus S_{N_1}$.

Because member $N_1N_2\ldots N_m$ knows $S_{\bar{N}_1}$, $S_{N_1\bar{N}_2}$, …, $S_{N_1N_2\ldots\bar{N}_{m-1}}$, these keys must be updated. The group key manager updates all the keys that must be updated and group key by Eqs. (3) and (4).

$$S'_{N_1N_2\ldots\bar{N}_n} = S_{N_1N_2\ldots\bar{N}_n} \oplus H^{m-n-1}(S_{N_1N_2\ldots N_m}) \quad (3)$$

where n is less than m. After keys tree is updated, the number of member $N_1N_2\ldots\bar{N}_m$ will change into $N_1N_2\ldots N_{m-1}$. $S_{N_1N_2\ldots\bar{N}_m}$ will not exist in the new keys tree. The new group key is

$$\mathrm{GK}' = \mathrm{GK} \oplus H^{m-1}(S_{N_1N_2\ldots N_m}) \quad (4)$$

All the members are classified into several classes by the number of members: the members whose numbers begin with $N_1N_2\ldots\bar{N}_m$, the members whose numbers begin with $N_1N_2\ldots\bar{N}_{m-1}$, …, the members whose numbers begin with $\bar{N}_1$.

If the node whose number is $N_1N_2\ldots\bar{N}_m$ is a leaf node, then this leaf node corresponds to a group member. This member performs the following operations: First, it computes $H^1(S_{N_1N_2\ldots N_m})$, …, $H^{m-1}(S_{N_1N_2\ldots N_m})$ by $S_{N_1N_2\ldots N_m}$ and one-way function H. Secondly, it computes $S'_{\bar{N}_1}$, $S'_{N_1\bar{N}_2}$, …, $S'_{N_1N_2\ldots\bar{N}_{m-1}}$ by Eq. (3) and new group key GK′ by Eq. (4). Finally, this member changes its number to $N_1N_2\ldots N_{m-1}$.

If the node whose number is $N_1N_2\ldots\bar{N}_m$ is not a leaf node, then all the members below this node (those nodes whose numbers begin with $N_1N_2\ldots\bar{N}_m$) can compute all the keys that must be updated and new group key by the known $S_{N_1N_2\ldots N_m}$. After they finish these jobs, these group members get rid of $\bar{N}_m$ in the numbers of nodes whose numbers begin with $N_1N_2\ldots\bar{N}_m$ (including member number).

For those members whose numbers begin with $N_1N_2\ldots\bar{N}_n$ ($0 < n < m$), they can obtain $H^{m-n}(S_{N_1N_2\ldots N_m})$ by known $S_{N_1N_2\ldots N_m}$ and $H^{m-n}(S_{N_1N_2\ldots N_m}) \oplus S_{N_1N_2\ldots N_m}$ which is included in the keys tree update message and compute $H^{m-n+1}(S_{N_1N_2\ldots N_m})$, …, $H^{m-1}(S_{N_1N_2\ldots N_m})$. Then they can compute $S'_{N_1N_2\ldots\bar{N}_{n-1}}$, …, $S'_{\bar{N}_1}$ and the group key GK′ by Eqs. (3) and (4). For those members whose numbers begin with $\bar{N}_1$, they only need to update the group key.

The changes in the keys tree is depicted in Fig. 3 when member 010 leaves the group.

$$S'_{00} = S_{00} \oplus S_{010}, \quad S'_1 = S_1 \oplus H^1(S_{010})$$
$$\mathrm{GK}' = \mathrm{GK} \oplus H^2(S_{010})$$

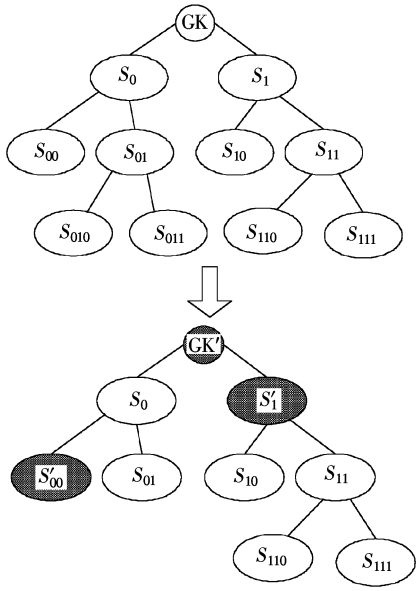

Fig. 3 Changes of keys tree when member 010 leaves the group

The group key manager will multicast the following keys tree update message: The number of the member is 010; $H^1(S_{010}) \oplus S_{01}$, $H^2(S_{010}) \oplus S_0$.

Members 00 can compute $H^1(S_{010})$ from its known S_{01} and $H^1(S_{010}) \oplus S_{01}$ which is included in the keys tree update message. By $H^1(S_{010})$ member 00 can compute $H^2(S_{010})$, so it can compute S'_1 and GK′.

Member 011 changes its number to 01. By its known S_{010} he can compute $H^1(S_{010})$. By $H^1(S_{010})$ he can compute $H^2(S_{010})$, so he can compute S'_1 and GK′.

As for members 10, 110 and 111, they can compute $H^2(S_{010})$ from known S_0 and $H^2(S_{010}) \oplus S_0$ which is included in the keys tree update message. So these members can compute the new group key GK′.

2 Security Analyses on New Group Key Management Scheme

In secure multicast communications a group key management scheme must present forward and backward security: Newly admitted group members cannot read previous messages and evicted members cannot read future messages, even with collusion by many arbitrarily evicted members. The following analysis indicates that our scheme can provide forward and backward security when the one-way function is secure.

First, we analyze the forward and backward security of our scheme when a single member leaves or joins the group.

After member $N_1N_2\dots N_m$ leaves the group, the group key will become $GK' = GK \oplus H^{m-1}(S_{N_1N_2\dots N_m})$. Because the member $N_1N_2\dots N_m$ does not know $S_{N_1N_2\dots N_m}$ and he cannot get $H^1(S_{N_1N_2\dots N_m})$, …, $H^{m-1}(S_{N_1N_2\dots N_m})$ from the keys tree update message, he cannot get the new group key GK'. At the same time all the keys which member $N_1N_2\dots N_m$ knows in the keys tree will be updated. These keys are $S_{N_1N_2\dots\bar{N}_m}$, $S_{N_1N_2\dots\bar{N}_{m-1}}$, …, $S_{\bar{N}_1}$. For member $N_1N_2\dots N_m$, he cannot get these new keys. He cannot get the following group keys from the following keys tree update messages, so backward security is ensured. When a new member joins into the group, the new group key is $GK' = GK \oplus H^m(\Delta S)$. Because the new member cannot get ΔS, $H(\Delta S)$, …, $H^m(\Delta S)$, he cannot get the foregoing group keys from this new group key and the keys tree update messages.

Secondly, we analyze the collusion attack. The aim of the collusion is to attain the group key when all the attackers are not group members. The main form of the collusion attack is of two kinds. One kind is that attackers want to get the group key after they all leave the group. The other kind is that attackers want to get the group key before they all join into the group. There is no meaning to consider collusion attacks that include current members because every member knows the current group key.

Our idea is to prove that, arbitrarily, many group members cannot get those keys which they should not know from all the keys they know in the keys tree and all the keys tree update messages which the group key manager uses to update the group key. By proving this we can ensure the security of our group key management scheme. We consider the collusion attack when the number of attackers is two.

Assume that the attackers are A, B whose member numbers are $P_1P_2\dots P_r0N_1N_2\dots N_t$ and $P_1P_2\dots P_r1M_1M_2\dots M_t$ (when the keys tree is not in balance, the proof is similar). Corresponding keys are X and Y. There are two kinds of collusion attacks. The first kind occurs when A and B are group members and they want to get the group key after they leave the group. The second kind occurs when A and B are not group members. They want to get the group key before they join into the group.

First we consider the first kind of collusion attack. The keys that member A knows are

$$S_{P_1P_2\dots P_r0N_1N_2\dots\bar{N}_t}, \dots, S_{P_1P_2\dots P_r0\bar{N}_1}, S_{P_1P_2\dots P_r1}, S_{P_1P_2\dots\bar{P}_r}, \dots, S_{\bar{P}_1}$$

The keys that member B knows are

$$S_{P_1P_2\dots P_r1M_1M_2\dots\bar{M}_t}, \dots, S_{P_1P_2\dots P_r1\bar{M}_1}, S_{P_1P_2\dots\bar{P}_r0}, \dots, S_{\bar{P}_1}$$

So all the keys member A and member B know are

$$S_{P_1P_2\dots P_r0N_1N_2\dots\bar{N}_t}, \dots, S_{P_1P_2\dots P_r0\bar{N}_1}, S_{P_1P_2\dots P_r1}, S_{P_1P_2\dots\bar{P}_r}, \dots, S_{\bar{P}_1}$$
$$S_{P_1P_2\dots P_r1M_1M_2\dots\bar{M}_t}, \dots, S_{P_1P_2\dots P_r1\bar{M}_1}, S_{P_1P_2\dots P_r0}$$

When member A leaves the group, the keys tree update message that the group key manager will multicast is

$$H(X) \oplus S_{P_1P_2\dots P_r0N_1N_2\dots N_{t-1}}, \dots, H^{t-1}(X) \oplus S_{P_1P_2\dots P_r0N_1},$$
$$H^t(X) \oplus S_{P_1P_2\dots P_r0}, H^{t+1}(X) \oplus S_{P_1P_2\dots P_r}, \dots, H^{t+r}(X) \oplus S_{P_1}$$

Because member A and member B do not know $S_{P_1P_2\dots P_r0N_1N_2\dots N_{t-1}}$, …, $S_{P_1P_2\dots P_r0N_1}$, they cannot get X, $H^1(X)$, …, $H^{t-1}(X)$. Due to

$$S'_{P_1P_2\dots P_r0N_1N_2\dots\bar{N}_{t-1}} = S_{P_1P_2\dots P_r0N_1N_2\dots\bar{N}_{t-1}} \oplus H^0(X)$$
$$S'_{P_1P_2\dots P_r0\bar{N}_1} = S_{P_1P_2\dots P_r0\bar{N}_1} \oplus H^{t-2}(X)$$
$$S'_{P_1P_2\dots P_r1} = S_{P_1P_2\dots P_r1} \oplus H^{t-1}(X)$$

so member A and member B cannot get $S'_{P_1P_2\dots P_r0N_1N_2\dots\bar{N}_{t-1}}$, …, $S'_{P_1P_2\dots P_r0\bar{N}_1}$, $S'_{P_1P_2\dots P_r1}$. Now all the keys that member A and member B know are

$$S'_{P_1P_2\dots\bar{P}_r}, \dots, S'_{\bar{P}_1}, S_{P_1P_2\dots P_r1M_1M_2\dots\bar{M}_t}, \dots,$$
$$S_{P_1P_2\dots P_r1\bar{M}_1}, S_{P_1P_2\dots P_r0}$$

In fact these keys are those that member B must know. That is to say, B cannot get more keys than those he must know by the information that A knows and the keys tree update message. After member B leaves the group, member A and member B will not know any keys in the keys tree and they cannot get the following group keys.

For the second kind of collusion attack we can arrive at the same result in the same way. For this kind of collusion attack the one-way function chains $H^1(\Delta S)$, …, $H^m(\Delta S)$ ensure the security. In the same way, we can prove these attackers cannot get any keys that they should not know when the number of collusion attackers is any.

3 Transmission, Storage and Computational Costs of New Group Key Management Scheme

Simple key distribution center(SKDC)[4,5] is the simplest group key distribution solution in which a group manager shares a secret key with each member and sequentially uses each member's key to communicate the secret group key to that member. Each time that member is added to (or evicted from) a group with n members, the group manager must perform n encryptions and transmit n keys. Our scheme is similar to those schemes which are based on the keys tree. Transmission cost is $O(\lg n)$ and the storage cost of every member is also $O(\lg n)$.

The numbers in Tab. 1 and Tab. 2 are approxi-

Tab. 1 Comparison of the maximum computational and transmission cost

Items	Evicting a member			Adding a member		
	Multicast size/bits	Manager computation	Maximum member computation	Multicast size/bits	Manager computation	Maximum member computation
SKDC	$nK + \lg n$	nC_E	C_E	$nK + \lg n$	$nC_E + C_r$	C_E
LKH	$2hK + h$	$h(2C_E + C_r)$	hC_E	$2hK + h$	$h(2C_E + C_r)$	hC_E
OFT	$hK + h$	$C_r + h(C_E + 2C_f)$	$h(C_E + C_f)$	$hK + h$	$h(C_E + 2C_f) + C_r$	$h(C_E + C_f)$
Our scheme	$hK + h$	hC_f	hC_f	$hK + h$	$hC_f + 3C_r$	hC_f

Tab. 2 Storage cost of the group manager and group members bit

Items	Manager storage	Member storage
SKDC	nK	$2K$
LKH	$2nK$	hK
OFT	$2nK$	hK
Our scheme	$2nK$	hK

mate. Where n is the number of members; h is the height of the keys tree; C_E, C_r, C_f are respectively the computational cost of one evaluation of the encryption function, generating one key from a cryptographically-secure random source, and one evaluation of the one-way function. The detailed way of computation of all kinds of costs is similar to the way which is presented in Ref. [7]. The numbers in Tab. 1 do not contain the cost for computing and transmitting the digital signature of the group manager.

Among all these schemes the performance of SKDC is the poorest. The transmission cost of our scheme is the same as the transmission cost of OFT. It is about one half of the transmission cost of LKH. As far as computation cost is concerned, our scheme has better advantages, because there are no encryption and decryption operations in these two schemes, which is not the case in LKH and OFT. Our scheme just needs simple XOR operations and one-way function computation. These are faster than encryption and decryption operations. All these schemes, except SKDC, are about equivalent in storage cost.

References

[1] Moyer Matthew J, Rao Josyula R, Rohatgi Pankaj. A survey of security issues in multicast communications [J]. *IEEE Network*, 1999, **13**(6): 12 - 23.

[2] Judge Paul, Ammar Mostafa. Gothic: a group access control architecture for secure multicast and anycast [C]// *IEEE Information and Communications Conference*. New York, 2002: 1547 - 1556.

[3] Stein M, Tsudik G, Waidner M. Key agreement in dynamic peer groups [J]. *IEEE Trans on Parallel and Distributed Systems*, 2000, **11**(8): 769 - 780.

[4] Harney H, Muckenhirn C, Rivers T. RFC2093: group key management protocol (GKMP) specification [R]. USA: IETF, 1997.

[5] Harney H, Muckenhirn C, Rivers C. RFC2094: group key management protocol (GKMP) architecture [R]. USA: IETF, 1997.

[6] Harney H, Harder E. Logical key hierarchy protocol [R]. USA: IETF, 1999.

[7] Sherman Alan T, McGrew David A. Key establishment in large dynamic groups using one-way function trees [J]. *IEEE Transactions on Software Engineering*, 2003, **29**(5): 444 - 458.

一种新的基于密钥树、XOR 操作及单向函数的组密钥管理方案

张 勇 张 翼 汪为农

(上海交通大学网络信息中心,上海 200030)

摘要:在基于密钥树的组密钥管理方案中引入了 XOR 操作以及单向函数链,提出了一种新的基于密钥树、XOR 操作以及单向函数链的组密钥管理方案,介绍了新方案中初始化、成员加入以及成员退出的操作,将新方案与 3 种基于密钥树的组密钥管理方案 SKDC,LKH,OFT 进行了比较,数据表明:就传输、计算以及存储开销而言,新的组密钥管理方案性能最优. 分析了新方案的安全性问题,该方案能保证前向以及后向安全性:新加入的组成员不能获得以前的组播报文,而且即使任意多个退出组播组的组成员进行合谋也不能获得以后的组播报文.

关键词:安全组播通信;组密钥管理;密钥树;单向函数

中图分类号:TP393.08