An group key distribution protocol for secure group communications

Wan Fang[1] , Wang Dazhen[1,2]
{1.School of Computer Science and Technology, Hubei University of Technology, Wuhan, P.R.C 430068}
{2.School of Information Management, Wuhan University, Wuhan, P.R.C 430074}

Abstract: How to implement each member of a communication group having the same group key is the base of secure group communication which is the main communication form in distributed system. Group key distribution protocol is the main method to make each member has same group key. This paper proposes a group key distribution protocol based on one-way key chain tree (OKCT). Firstly, we propose the definition of secure one-way key chain tree. Secondly, we respectively propose the protocol of the group key distribution when member joins or leaves the group. Thirdly, we prove the security of group distribution protocol based OKCT. Finally, we analyze the performance of group distribution protocol based OKCT.

Key-Words: Group Communication, Secure Communication, Key Distribute, OKCT, HOKCT

1 Introduction

Group communication is the main communication form in distributed system, such as distributed compute, grid compute, muli-agent communication, live multiparty conferencing, and network game. How to make the group communication in security becomes more and more important. However, compared with the mature security mechanism of point to point communication, there are many security problems in group communication to be solved[1,2,3,4]. The foremost problem is how to make all members of a group have the same group key. The member can use the same group key to encrypt and authenticate the group communication message, and ensure the confidentiality, integrity and authenticity of group communication message.

Group key distribution protocol is the main method to make each member has same group key. Group key distribution protocol needs a Group Control Key Server (GCKS) to compute and distribute group key. GKCS also needs re-key group key with member joining or leaving group dynamically. The base performance goal of group key distribution protocol is the cost for communication, computation and storage as little as possible.

The Group Key Management Protocol (GKMP) is pioneer group key distribution protocol. There has been corresponding architecture[5] and specification[6] in 1997. GKMP has the advantages of principle simpleness and implement easiness. But the most disadvantage of GKMP is that the computation cost and communication cost which member leaving re-key protocol needs are proportional to the number of group member. The scalability of GKMP is so bad that GKMP cannot suit to group communication application with large numbers of members.

In order to improve the scalability of group key distribution protocol, researches proposed many protocols based key tree[7-13]. Wallner et al [7] and Caronni et al [8,9] propose the group key distribution protocol based a Hierarchical Binary Tree (HBT). In this protocol, the GCKS maintains a tree of keys. Each node of the key tree holds one and only key, and the leaf node corresponds to a member. Each member receives and maintains a copy of the all keys held by each node in the path from the corresponding leaf node to the root node. The key held by the root node is the group key. Through maintains the keys as a HBT, this protocol decreases the computation cost and communication cost to $O(2\log_2 n)$. Wong et al [10] and Canetti et al [11] extend the binary tree to a k-ary tree. A tree with a

larger degree reduces the number of keys kept by members and GCKS.

The group key distribution protocol based a one-way function tree (OFT)[12,13] proposed by Balenson and McGrew improves the performance of HBT. Their scheme reduces the communication quantity to $O(\log_2 n)$. Each node of OFT hold not only an original key but also a blind key. The blind key is computed by a one–way function to the original key. Each member maintains the key held by the corresponding leaf key and the blind keys held by the sibling nodes of all the nodes in the path from the corresponding leaf node to the root node.

This paper proposes a group key distribution protocol based One-way Key Chain Tree (OKCT). The principle of OKCT is the updated keys form a one-way key chain when member joins or leaves group. OKCT requires each member decrypting just one re-key message to update group key, and decreases the computing quantity of each member. OKCT also decreases the communication quantity, and do not decrease the security.

2. One-way Key Chain Tree

One-way key chain has been applied in many cryptography fields, such as S/Key[14] and TESLA[15]. If we can easily compute the keys of a key list $k_1,k_2,\ldots,k_t$ with the former keys, but cannot compute the keys with the latter keys, we regard the key list $k_1,k_2,\ldots,k_t$ as a one-way key chain.

Definition 1（**Secure One-way Key Chain**） A key list $k_1,k_2,\ldots,k_t$ is a secure one-way key chain, if for any probabilistic polynomial time attack algorithm $\mathcal{F}$, any polynomial p, and all sufficiently large n, the key list $k_1,k_2,\ldots,k_t$ satisfies the following two conditions.

（1） $\forall i,j\in\{1,2,\ldots,t\}, i>j \ \exists\mathcal{F}$， $\mathcal{F}(k_j)=k_i$

（2） $\forall i,j\in\{1,2,\ldots,t\}, i>j \ \forall\mathcal{F}$， $\Pr[\mathcal{F}(k_i)=k_j]<\frac{1}{p(n)}$

If denoting f as a one-way function and f^{j-i} as iteratively computing f $j-i$ times, we can construct a secure one-way key chain with an initial key k_0 as follows.

$$\forall i,j\in\{1,2,\ldots,t\}, i<j \ ， \ k_j=f^{j-i}(k_i)$$

Definition 2（**One-way Key Chain Tree**） A key tree T is an one-way key chain tree, if there is a leaf node that the key list held by the nodes in the path from the leaf node to the root node is a secure one-way key chain.

In figure 1, the key tree is a one-way key chain tree because the key list k_6,k_2,k_0 is a secure one-way key chain.

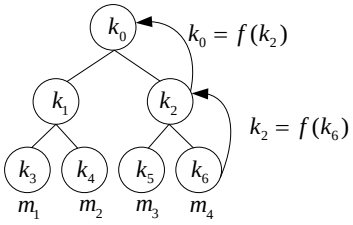

Fig 1 one-way key chain tree

3. Group Key Distribution protocol Based OKCT

We use the following notations to illustrate our protocol as table 1.

Tab.1. Notations and their meanings

T, T^+, T^-	Current key tree, the key tree after member joining, the key tree after member leaving.
k	Degree of key tree.
$N, IN, LN, \overline{N}, N_r$	Nodes of key tree. N_r denotes the root node， LN denotes leaf node.
k_N, k_N^+, k_N^-	Keys held by node N of T, T^+, T^-.
M^T	Member set corresponding to the leaf nodes of T.
K^T	Key set of T.
$keyset^T(m)$	Key set member m maintains in T.
$keyset^T(\overline{M})$	Intersection key set $m \in \overline{M}$

	maintains in T where $\bar{M} \subseteq M$.
$memset^T(k)$	Member set who maintain k in T.
$memset^T(\bar{K})$	Intersection member set who maintain $k \in \bar{K}$ in T where $\bar{K} \subseteq K$.
$okckset^T()$	One-way key chain of T.
$parent^T(N)$,	Parent node of N in T.
$sibling^T(N)$	Sibling node of N in T.
$children^T(N)$	Children nodes of N in T.
$ancestor^T(N)$	Node set in the path from N to the root node in T.
$GCKS \rightarrow x:y$	GCKS send x to y
$E_{k_1}(k_2)$	Encryption k_2 with k_1
$\Pr[g(x) \mid x \leftarrow \{x_1, x_2, ..., x_n\}]$	probability when $g(x)$ is true where x is selected randomly from $x_1, x_2, ..., x_n$

3.1 Group Key Distribution protocol when member joins group

When a member m requests to join group, GCKS authenticates whether m is entitled to join group. If m can join group, GCKS performs group key distribution protocol as follows.

(1) News a leaf node corresponding to m, and denotes the leaf node as LN.

(2) Generates k_{LN} and sends k_{LN} to m in a secure method.

(3) Finds a node N to insert LN. If there is a node N whose children node number is less than k, the node N is the anticipant node. GCKS set LN as a children node of N to insert LN as figure 2(a). Else the highest (nearest to root node) and most left leaf node is N. In order to insert LN, GCKS news a node IN and sets N and LN as the children nodes of IN as figure 2(b).

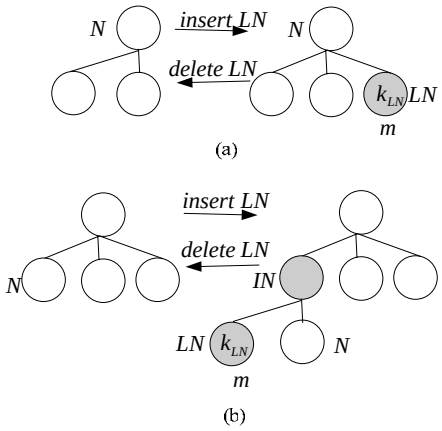

Fig.2. insert and delete leaf node

(4) Computes the new secure one-way key chain as follows.

$$\forall \bar{N} \in ancestor^{T^+}(LN) - \{N_r\} + \{LN\}, \quad k^{+}_{parent^{T^+}(\bar{N})} = f(k^{+}_{\bar{N}})$$

(5) Distributes new keys to corresponding members. If GCKS hasn't generated new node IN when insert LN, GCKS encrypts the new keys in T^+ with corresponding old key in T and sends re-key message to corresponding members as follows.

$$\forall \bar{N} \in ancestor^{T^+}(LN) - \{N_r\} + \{LN\},$$
$$GCKS \rightarrow memset^{T^+}(k^{+}_{parent^{T^+}(\bar{N})}) - memset^{T^+}(k^{+}_{\bar{N}}) : E_{k^{+}_{parent^{T^+}(\bar{N})}}(k^{+}_{parent^{T^+}(\bar{N})})$$

Else GCKS yet encrypts k_{IN} with k_N and sends the re-key message to the all members who maintains k_{IN} as follows.

$$GCKS \rightarrow memset^{T^+}(k_{IN}) : E_{k_N}\{k_{IN}\}$$

Member decrypts received re-key message and computes secure one-way key chain to get the new group key as follows.

(1) m performs iteratively one-way function from k_{LN} to get new secure one-way key chain.

(2) The other member decrypts received re-key

message to get new key and performs iteratively one-way function from the new key to get other new required keys.

Figure 3 gives an example to illustrate group key distribution when member joins group. GCKS renews the key tree and distributes new keys when member m_{10} joins group as follows.

(1) News a leaf node LN corresponding to m_{10}.

(2) Generates k_{LN} and sends k_{LN} to m in a secure method.

(3) News a node IN, replaces N_4 with IN, and sets LN and N_4 as the children nodes of IN. Where N_4 is the highest and most left leaf node.

(4) Computes new secure one-way key chains $k_{LN}, k_{IN}, k_1^+, k_0^+$ as follows: $k_{IN}=f(k_{LN})$, $k_1^+=f(k_{IN})$, $k_0^+=f(k_1^+)$.

(5) Sends $E_{k_4}(k_{IN})$ to m_1, sends $E_{k_1}(k_1^+)$ to $\{m_2,m_3\}$, and sends $E_{k_0}(k_0^+)$ to $\{m_4,m_5,...,m_9\}$.

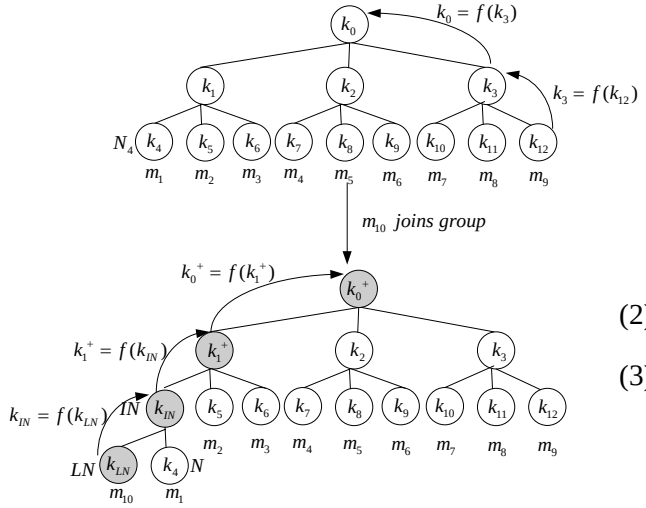

Fig.3. renew one-way key tree when m_{10} joins

Member decrypts received re-key message and computes secure one-way key chain to get the new group key as follows.

(1) m_{10} computes new secure one-way key chain $k_{LN}, k_{IN}, k_1^+, k_0^+$ from k_{LN} as follows: $k_{IN}=f(k_{LN})$, $k_1^+=f(k_{IN})$, $k_0^+=f(k_1^+)$.

(2) m_1 decrypts re-key message $E_{k_4}(k_{IN})$ to get k_{IN} and computes one-way key chain k_{IN}, k_1^+, k_0^+ from k_{IN} as follows: $k_1^+ = f(k_{IN})$, $k_0^+ = f(k_1^+)$.

(3) $m \in \{m_2, m_3\}$ decrypts re-key message $E_{k_1}(k_1^+)$ to get k_1^+ and computes one-way key chain k_1^+, k_0^+ as $k_0^+ = f(k_1^+)$.

(4) $m \in \{m_4, m_5, ..., m_9\}$ decrypts re-key message $E_{k_0}(k_0^+)$ to get k_0^+.

3.2 Group Key Distribution protocol when member leaves group

When a member m leaves group, GCKS performs group key distribution protocol as follows.

(1) Deletes the leaf node LN corresponding to m. If $parent^T(LN)$ children number is big than 2, GCKS just deletes LN as figure 2(a). If $parent(LN)$ children number is equal to 2, GCKS should replaces $parent^T(LN)$ with $silbing^T(LN)$ after deleting LN as figure 2(b).

(2) Renews k_N to k_N^-.

(3) Computes new secure one-way key chains as follows.

$$\forall \overline{N} \in ancestor^{T^-}(N) - \{N_r\} + \{N\},$$
$$k_{parent^{T^-}(\overline{N})}^- = f(k_{\overline{N}}^-)$$

(4) Distributes new keys held by nodes in the path from N to N_r as follows.

$$\forall \overline{N} \in ancestor^{T^-}(N) - \{N_r\} + \{N\},$$

$$\forall \overline{N}_S \in sibling^{T^-}(\overline{N}),$$

$$GCKS \rightarrow memset(k_{\overline{N}_S}) : E_{k_{\overline{N}_S}}(k_{parent^{T^-}(\overline{N})})$$

$$\forall \overline{N} \in ancestor^{T^-}(N) + \{N\} - \{N_r\},\ k_{parent^{T^-}(\overline{N})} = f(k_{\overline{N}}^{-})$$

Member decrypts received re-key message and computes secure one-way key chain to get the new group key after receiving re-key message.

Figure 4 gives an example to illustrate group key distribution when member leaves group. GCKS renews the key tree and distributes new keys when member m_9 leaves group as follows.

(1) Deletes leave node corresponding to m_9.

(2) Renews k_3 to k_3^-.

(3) Computes secure one-way key chain k_3^-, k_0^- as $k_0^- = f(k_3^-)$.

(4) Sends $E_{k_7}(k_3^-)$ and $E_{k_8}(k_3^-)$ to $\{m_7, m_8\}$, sends $E_{k_1}(k_0^-)$ to $\{m_1, m_2, m_3\}$ and sends $E_{k_2}(k_0^-)$ to $\{m_4, m_5, m_6\}$.

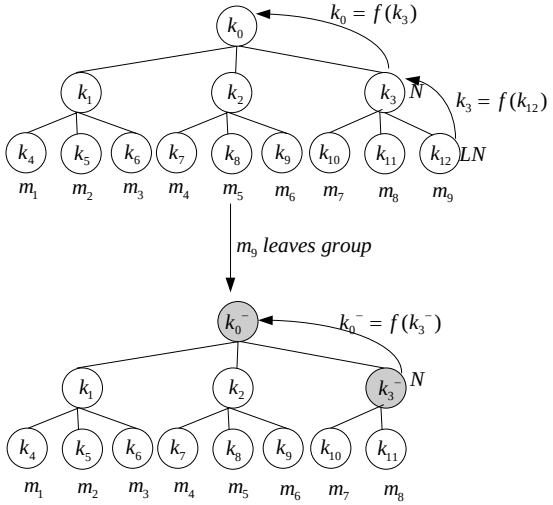

Fig.4. renew one-way key tree when m_9 leaves

Member decrypts received re-key message and computes secure one-way key chain to get the new group key as follows.

(1) m_7 decrypts $E_{k_7}(k_3^-)$ to get k_3^-. m_8 decrypts $E_{k_8}(k_3^-)$ to get k_3^-. $m \in \{m_7, m_8\}$ computes new one-way key chain k_3^-, k_0^- as $k_0^- = f(k_3^-)$.

(2) $m \in \{m_1, m_2, m_3\}$ decrypts $E_{k_1}(k_0^-)$ to get k_0^-. $m \in \{m_4, m_5, m_6\}$ decrypts $E_{k_2}(k_0^-)$ to get k_0^-.

4 Proofs of security

Definition 3 (Secure Key Tree) A key tree T is a secure key tree, if for any probabilistic polynomial time attack algorithm $\mathcal{F}$, any polynomial p, and all sufficiently large n, each member(m) of group can get other keys which don't be maintained by m with probabilistic less than $\frac{1}{p(s)}$:

$$\forall m \in M^T,\ \Pr[k \in K^T - keyset^T(m) \mid k \leftarrow \mathcal{F}(keyset^T(m))] < \frac{1}{p(n)}$$

Definition 4 (Perfect Backward Secrecy Group Key Distribution protocol) A secure group key distribution protocol is perfect backward secrecy, if for any probabilistic polynomial time attack algorithms $\mathcal{F}$, any polynomial p, and all sufficiently large n, the new member(m) can get keys in old key tree (T) with probabilistic less than $\frac{1}{p(n)}$ even when m receives all re-key message and the new key tree T^+ is secure key tree. That is the protocol satisfies the following conditions.

(1) $\Pr[k \in K^T \mid k \leftarrow \mathcal{F}(keyset^{T^+}(m), allrekeym())] < \frac{1}{p(n)}$。

（2） T^+ is secure key tree.

Definition 5 (Perfect Forward Secrecy Group Key Distribution protocol**)** A secure group key distribution protocol is perfect forward secrecy, if for any probabilistic polynomial time attack algorithms $\mathcal{F}$, any polynomial p, and all sufficiently large n, the departed member(m) can get keys in new key tree (T^-) with probabilistic less than $\frac{1}{p(n)}$ even when m receives all re-key message and the new key tree T^- is secure key tree. That is the protocol satisfies the following conditions.

（1） $\Pr[k \in K^{T^-} \mid k \leftarrow \mathcal{F}(keyset^{T}(m), allrekeym())] < \frac{1}{p(n)}$

（2） T^- is secure key tree.

Definition 6 (Secure Group Key Distribution protocol) A secure group key distribution protocol is secure, if the protocol is perfect backward Secrecy and perfect forward secrecy.

Theorem 1 Suppose the encryption function E is security against adaptive chosen message attack, the group key distribution protocol based OKCT is perfect backward secrecy.

Proof: (1) Because E is security against adaptive chosen message attack, the following expression is true, for any probabilistic polynomial time attack algorithms $\mathcal{F}$, any polynomial p, and all sufficiently large n.

$$\Pr[k \in kekset^{T}() \mid k \leftarrow \mathcal{F}(\{E_{k_1}(k_2)\}), k_2 \in keyset^{T^+}(m)] < \frac{1}{p(n)}$$

That is the following expression is true.

$$\Pr[k \in kekset^{T}() \mid k \leftarrow \mathcal{F}(keyset^{T^+}(m), allrekeym())] < \frac{1}{p(n)}$$

The new member m cannot get any key $k \in K^{T} - kekset^{T}()$ from $keyset^{T^+}(m)$ and $allrekeym()$, so the following expression is true.

$$\Pr[k \in K^{T} \mid k \leftarrow \mathcal{F}(keyset^{T^+}(m), allrekeym())] < \frac{1}{p(n)}$$

（2）Apparently, the following expression is true.

$$\forall \bar{m} \in M^{T^+} - \{m\}, \quad \Pr[k \in keyset^{T^+}(\bar{m}) - keyset^{T^+}(m) \mid k \leftarrow \mathcal{F}(keyset^{T^+}(m))] < \frac{1}{p(n)} \qquad (a)$$

Because $keyset(m)^+$ is secure one-way key chain, the following expression is true.

$$\Pr[k \in keyset^{T^+}(m) - keyset^{T^+}(\bar{m}) \mid k \leftarrow \mathcal{F}(keyset^{T^+}(\{\bar{m}, m\})] < \frac{1}{p(n)}$$

$keyset^{T^+}(\bar{m}) - keyset^{T^+}(m)$ is independent of $keyset^{T^+}(m) - keyset^{T^+}(\bar{m})$, so the following expression is true.

$$\Pr[k \in keyset^{T^+}(m) - keyset^{T^+}(\bar{m}) \mid k \leftarrow \mathcal{F}(keyset^{T^+}(\bar{m})] < \frac{1}{p(s)} \qquad (b)$$

According to the two expressions (a) and（b）, we can conclude the following expression is true.

$$\forall m \in M^{T^+}, \quad \Pr[k \in K^{T^+} - keyset^{T^+}m) \mid k \leftarrow \mathcal{F}(keyset^{T^+}(m))] < \frac{1}{p(n)}$$

So T^+ is a secure key tree.

According to (1)(2), we can conclude that the group key distribute protocol is perfect backward secrecy.

Theorem 2 Suppose the encryption function E is security against adaptive chosen message attack, the group key distribution protocol based OKCT is perfect forward secrecy.

Proof:（1）Because E is security against adaptive chosen message attack, the following expression is true, for any probabilistic polynomial time attack algorithms $\mathcal{F}$, any polynomial p, and all sufficiently large n.

$$\Pr[k \in okckset^{T^-}() \mid k \leftarrow \mathcal{F}(\{E_{k_1}(k_2)\}), k_2 \in okckset^{T^-}()] < \frac{1}{p(n)}$$

That is the following expression is true.

$$\Pr[k \in okckset^{T^-}() \mid k \leftarrow \mathcal{F}(keyset(m), allrekeym())] < \frac{1}{p(s)}$$

Apparently, the departed member m cannot get key $k \in K^{T^-} - okckset^{T^-}()$ from $keyset(m)$ and $allrekeym()$, so the following expression is true.

$$\Pr[k \in K^{T^-} \mid k \leftarrow \mathcal{F}(keyset^{T}(m), allrekeym())] < \frac{1}{p(n)}$$

（2）Because the key set $keyset^{T^-}(m)$ as secure one-way key chain, we can proof the following expression is true as proofing theorem 1.

$$\forall m \in M^{T^+},\ \Pr[k \in K^{T^+} - keyset^{T^+} m) \mid k \leftarrow \mathcal{F}(keyset^{T^+}(m))] < \frac{1}{p(n)}$$

So T^- is a secure key tree.

According to (1)(2), we can conclude that the group key distribute protocol is perfect forward secrecy.

Corollary 1: Suppose the encryption function E is security against adaptive chosen message attack, the group key distribution protocol based OKCT is secure.

5 Performance Analyses

Now we analyze the performance of group key distribution protocol based on OKCT through comparing with other popular protocols in storage cost, communication cost and computation cost factors. In order to facilitate comparing among protocols, we suppose that each key trees is a full tree before member joins and leaves group.

5.1 Storage Cost Analyses

Storage cost includes the keys GCKS must maintain and member must maintain. Table 2 lists the storage cost of each protocol. According to table 2, GKMP just need GKCS maintain $n+1$ keys and member maintain 2 keys. Although key tree decreases communication cost and computation cost, key trees increase storage cost. The storage cost is increases with the key degree k decrease. The storage cost of HBT and OFT is biggest. The storage cost of OKCT is same as of HKT.

Tab.2 Storage cost comparison

Protocols	Key number	
	GCKS maintains	Member maintains
GKMP	$n+1$	2
HBT	$2n-1$	$\log_2 n+1$
OFT	$2n-1$	$\log_2 n+1$
HKT	$\frac{kn-1}{k-1}$	$\log_k n+1$
OKCT	$\frac{kn-1}{k-1}$	$\log_k n+1$

5.2 Communication Cost Analyses

The communication cost when member joins or leaves group is the main factor to estimate protocol performance, especially for wireless group communication with lacking bandwidth. Because each re-key message is composed of encrypted key with same length, the communication cost is proportional to the re-key message number. Table 3 lists the communication cost of each protocol.

Tab.3 communication cost comparison

Protocols	Re-key messages number	
	Member joins	Member leaves
GKMP	2	$n-1$
HBT	$2\log_2 n+2$	$2\log_2 n-1$
OFT	$2\log_2 n+2$	$\log_2 n-1$
HKT	$2\log_k n+2$	$k\log_k n-1$
OKCT	$\log_k n+1$	$(k-1)\log_k n$

According to table 3, we can conclude that GKMP need just two re-key messages when new member joins group, but GKMP need $n-1$ re-key messages when member leaves group, which makes GKMP cannot be applied in group communication with large numbers of members. When member joins group, OKCT need just $\log_k n+1$ re-key messages, which is less than the number of re-key messages in HBT, OFT, and HKT. When member leaves group, OKCT need just $(k-1)\log_k n$ re-key messages, which is also less than the number of re-key messages in HBT, OFT, and HKT.

5.3 Computation Cost Analyses

The main computation cost of GCKS is encrypting re-key messages, and the main cost of member is decrypting re-key messages. Because the computation cost of different member is different, we analyze the max and average computation cost.

The computation cost of OFT and OKCT also includes performing one-way function.

Table 4 lists the communication cost of each protocol when member joins group, where T_c, T_d, T_h respectively denotes the time to perform one-time encryption function, decryption function, and one-way function. Because $T_c >> T_h$ and $T_d >> T_h$, T_h can be omitted sometimes.

Tab.4. Computation cost comparison when member joins

Protocols	GCKS need	Member needs	
		greatest	average
GKMP	$2T_c$	T_d	T_d
HBT	$(2\log_2 n+2)*T_c$	$(\log_2 n+1)*T_d$	$\approx 2*T_d$
OFT	$(2\log_2 n+2)*(T_c+T_h)$	$(\log_2 n+1)*(T_d+T_h)$	$\approx 2*(T_d+T_h)$
HKT	$(2\log_k n+2)*T_c$	$(\log_k n+1)*T_d$	$\approx \frac{k}{k-1}*T_d$
OKCT	$(\log_k n+1)*(T_c+T_h)$	$T_d+(\log_k n+1)*T_h$	$\approx T_d+T_h$

According to table 4, we can conclude that the computation cost GCKS need in OKCT is $(\log_k n+1)*(T_c+T_h)$, which is less than the computation cost in HBT, OFT and OKCT. Specially, OKCT need member perform just one-time decryption function which decreases the computation cost largely.

Table 5 lists the communication cost of each protocol when member leaves group. According to table 5, we can conclude that GKMP need perform $n-1$ times encryption function, which will exhaust the computation resource. The computation cost GCKS need in OKCT is $(k-1)\log_k n*T_c+(\log_k n-1)*T_h$, which is less than the computation cost in HBT, OFT and OKCT. Specially, OKCT also need member perform just one-time decryption function.

Tab.5.Computation cost comparison when member leaves

Protocols	GCKS need	Member needs	
		greatest	average
GKMP	$(n-1)T_c$	T_d	T_d
HBT	$(2\log_2 n-1)*T_c$	$\log_2 n*T_d$	$\approx 2*T_d$
OFT	$(\log_2 n-1)*(T_c+T_h)$	$\log_2 n*(T_d+T_h)$	$\approx 2*(T_d+T_h)$
HKT	$(k\log_k n-1)*T_c$	$\log_k n*T_d$	$\approx \frac{k}{k-1}*T_d$
OKCT	$(k-1)\log_k n*T_c$ $+(\log_k n-1)*T_h$	$T_d+(\log_k n-1)*T_h$	$\approx T_d+T_h$

6 Conclusions an Future Work

The group key distribution based OKCT proposed in this paper is secure against adaptive chosen message attack, and has the properties of low communication cost and computation cost. Especially, our protocol just need member perform one time decryption function to renew group key, which decreases member computation cost largely. Our protocol is very suit be used in group communication application where terminal's computation resource is very lacking, such as mobile phone, PDA, and sensor.

Considering the different probability of member leaving group, we can make OKCT as a Huffman tree, and put the node corresponded to the member with high leaving group probability in lower layer. Huffman OKCT will make the average compute cost and communication cost much less when member leaving group probability is very different.

References:

[1] Y. Sun and K. Liu. Securing dynamic membership information in multicast communications. *IEEE Infocom*, 2004.

[2] D.A. Agarwal, O. Chevassut, M.R. Thompson, and G. Tsudik, An Integrated Solution for Secure Group Communication in Wide-Area Networks, *Proceedings of the 6th IEEE Symposium on Computers and Communications*,

Hammamet, Tunisia, July 3-5, 2001, pp 22-28.

[3] Hao-Hua Chu, Lintian Qiao, and Klara Nahrstedt, A Secure Multicast Protocol with Copyright Protection, *ACM SIGCOMM Computer Communications Review*, VOL.32, NO.2, 2002, pp.42-60.

[4] S. Banerjee and B. Bhattacharjee, Scalable secure group communication over IP multicast, *IEEE J-SAC*, VOL.20, NO.8,2002, PP.1511-1527.

[5] H. Harney and C. Muckenhirn, Group Key Management Protocol (GKMP) Architecture, *Internet Engineering Task Force*, July 1997. RFC 2094.

[6] H. Harney and C. Muckenhirn, Group Key Management Protocol (GKMP) Specification, *Internet Engineering Task Force*, July 1997. RFC 2093.

[7] D. Wallner, E. Harder and R. Age, Key Management for Multicast: Issues and Architectures, *Internet Engineering Task Force*, June 1999. RFC 2627.

[8] G. Caronni, M. Waldvogel, D. Sunand, and B. Plattner, Efficient Security for Large and Dynamic Multicast Groups. In *Workshop on Enabling Technologies, (WETICE 98)*. IEEE Comp Society Press, 1998.

[9] M. Waldvogel, G. Caronni, D. Sun, N. Weiler, and B.Plattner, The VersaKey Framework: Versatile Group Key Management. *IEEE Journal on Selected Areas in Communications*, VOL.17, NO.8, 1999, pp.1614-1631.

[10] C. K. Wong, M. G. Gouda, and S. S. Lam, Secure Group Communications Using Key Graphs, *IEEE/ACM Transactions on Networking*, VOL.8, NO.1, 2000, pp.16-30.

[11] R. Canetti, T. Malkin, and K. Nissim. Efficient Communication-Storage Tradeoffs for Multicast Encryption. In *Eurocrypt*, 1999.

[12] D. Balenson, D. McGrew, and A. Sherman, Key Management for Large Dynamic Groups: One-Way Function Trees and Amortized Initialization. *IETF Internet draft*, August 2000.

[13] A. T. Sherman and D. A. McGrew, Key establishment in large dynamic groups using one-way function trees, *IEEE Transactions on Software Engineering*, VOL.29, NO.5,2003, pp. 444-458.

[14] N.M. Haller, The S/KEY one-time password system, *Internet Soc. Symp. on Network and Distribute System*, 1994.

[15] A. Perrig, R. Canetti, D. Song, and J.D.Tygar, Efficient and Secure Source Authentication for Multicast. *In Symposium on Network and Distributed Systems Security (NDSS 2001)*, San Diego, CA, Feb, 2001.