Group Key Management Using Public Key Exchange

Babak Daghighi, Laiha Mat Kiah, and Saeedeh Afkhami Rad

Abstract—Group based applications have gained popularity in recent years. As these applications typically involve communication over open network, security is an important requirement. Group key management is one of main block in securing group communication. Though, centralized and distributed group key schemes rely on at least one explicit key manager for generating traffic key, contributory schemes employ the contribution of all members of a group for initiating new traffic key. This work presents an overview of existing contributory group key management schemes which specifically employ public key exchange algorithms for generating traffic key among members of a group. The initial operation which is used to generate first traffic key among members of group is demonstrated. Subsequently, required operations for provision of backward and forward secrecy during upon any changes in group membership are explored. And finally, the investigated schemes are analyzed and then compared in term of their communication cost.

Index Terms—Group communication, group key management, cryptographic algorithms

I. INTRODUCTION

In recent years, many group based applications such as multicast conferencing, information dissemination service, satellite TV distribution services and, more generally, applications supporting collaborative work have significantly gained in popularity. Since most of group based applications take place over insecure network, basic security services such as confidentiality, data integrity and entity authentication are necessary for them. These services can be established by a shared common key between entities in group. This key is used to encrypt all traffic that destined to a particular group. As a result, only members of the group can decrypt the received message.

One of fundamental challenges in designing secure and reliable group communication system is to manage a group key. There are several group key management schemes for disseminating group key to members of a group which can be classified into three categories as: 1) Centralized 2) Distributed 3) Contributory.

Centralized group key management involves a single entity (i.e. a group controller GC) which is responsible to generate, distribute and update the group key. One of the famous schemes in this category Logical Key Hierarch (LKH) that was proposed by several research groups nearly at the same time [1], [2]. Other existing approach can be found in, [3], [4], [5]. This approach has two problems: 1) dependencies on a key server leave a single point of failure and 2) it must be constantly available and present during group operation.

In another approach, distributed or also known as decentralized group key management, a large group is split into small subgroups. Each subgroup has its own subgroup controller which is responsible for key management in its subgroup. The first scheme was IOLUS [6] and then followed by some improvements schemes such as [7]-[9].

This approach can minimize the problem of concentrating on a single entity as a key server.

In contrast to these approaches, contributory group key management has no explicit key distributor center and all members contribute to manage the key(s) generation. This scheme helps to uniform distribution of the work load for key management and eliminates the need for central entity. This approach alleviates the problem of single points of failures in centralized group key management. Some contributory group key management schemes have been presented in [10], [11], [12], [13].

The goal of this paper is providing an overview of existing contributory group key management schemes. We will demonstrate the mechanisms which are used by these protocols that prevent a new joining member to access previous information as well as a leaving member to access future data communication within group communication in order to provide backward and forward secrecy.

The protocol that have been selected for analysis are ING, GDH1, GDH2, GDH3 [13], [14], Octopus [15], STR [11], [16], TGDH [10], [17] and CRTDH [18], which present variation of n-party Diffie Hellman group key agreement schemes.

The rest of paper is organized as follows: section II introduces public two party key exchanges. The group key management schemes are presented in section III. Section IV addresses a qualitative comparison of the discussed protocol. And finally, conclusion is presented in Section V.

II. PUBLIC TWO PARTY KEY EXCHANGE

A two party key exchange protocol is a protocol that allows two entities who have not contacted each other before, publicly exchange their information so that at the end of protocol they have a same share secret key. Any eavesdropper that listens to communication is unable to obtain this key. One of the first published public key crypto algorithm was Diffie Hellman (DH) [19].

If two entities are interested in using the DH for secure communication, they agree on Diffie Hellman group, a large prime number p and a generator number g. Next, the two entities choose secret private key xa and xb. They compute the public (blinded) key ya and yb by using secret private key and send them to each other.
A → B: $y_a = g^{x_a} \mod p$  
B → A: $y_b = g^{x_b} \mod p$

Now, they can compute secret share key by using partner’s public key 

$$K = (y_b)^{x_a} \mod p = (y_a)^{x_b} \mod p = g^{x_a \cdot x_b} \mod p$$

III. CONTRIBUTORY GROUP KEY SCHEMES

For simplicity, the following notations are used in this section:

- $M_i (1 \leq i \leq n)$: Group members;
- $p$: Large prime;
- $g$: A primitive root or generator of $\mathbb{Z}_p^*$;
- $x$: Denotes a random integer or secret exponent;
- $<L, i>$: The $i$th node at level $L$ in the binary key tree;
- $K_{<L, i>}$: The secret key of the node $<L, i>$;
- $b_{<L, i>}$: Blinded key of the node $<L, i>$;
- $b_i$: Blinded key of the $i$th group member;
- $K_i$: Secret key of the member $M_i$;
- CRT: Chinese Remainder Theorem;
- LCM: Least Common Multiple.

A. ING

The earliest attempt for extending two parties Diffie Hellman key exchange to a group has been done by Ingemarsson et al. (1982), and is called ING [12]. This protocol consists of (n-1) round and group members perform every round in synchronization. The member must be arranged in logical ring.

During the first round, each member $M_i$ computes $g^{x_i}$ and forwards it to its next neighbor or member $M_{i+1}$. In next rounds, each member raises the previously received intermediate key value to the power of its own exponent and forwards result to its logical neighbor. After (n-1) rounds, all participants possess a same key $K_n$.

B. GDH1

GDH1 [13] consists of two stages: upflow and downflow. The purpose of upflow stage is to collect contribution from all group members. In round $i$, each member $M_i$ receives a collection of intermediate value from $M_{i-1}$. Member $M_i$ raises the last value in received collection, i.e. $g^{x_1 \cdot x_{i-1}}$, to the power of its secret component $x_i$ and computes $g^{x_1 \cdot x_i}$. After that member $M_i$ appends new intermediate key value to the data of its own exponent and forwards result to its logical neighbor. The intended group key $K_n$ will be produced when the highest numbered group member computes $g^{x_1 \cdot x_n}$.

In downflow stage, member $M_i$ receives an ordered list of intermediate values from member $M_{i+1}$. It raises all values to the power of its secret component $x_i$. The last value in the set of computed intermediate values is $K_n$. Other values are sent to the member $M_{i+1}$.

C. GDH2

GDH2 generates shared key in $n$ rounds [13], [14]. The protocol collects contribution from group members in upflow stage. In this stage, each member $M_i$ receives a sequence of intermediate values and a cardinal value from $M_{i-1}$. The $M_i$ raises these values to its secret component $x_i$ and sends them to $M_{i+1}$.

In this protocol, the highest indexed group member $M_n$ plays the role of group controller. It receives the some values and raises them to its secret component. It keeps a cardinal value as an actual group key and broadcasts other intermediate values to the entire group. Each member $M_i$ recognizes its intermediate value from the receiving values and powers it by its $x_i$. As a result it computes the final group key.

When a new member is interested in joining to the group, the group controller $M_n$ generates a new secret component $x_n$ and creates a new upflow message by raising the contents of the previously received upflow message. It then sends the message to new member. The role of group controller is thus passed to new member of group.

In leave event, the member $M_n$ generates a new secret component $x_n$. Now, the $M_n$ compute a new set of $n-2$ sub keys (exclude sub key of leaving member $M_p$ and $M_q$) and broadcast them to all members of group. Since, $g^{x_1 \cdot x_p - 1} \cdot x_q \cdot x_{n-1}$ is missing from the set of broadcast sub keys, the departure member $M_p$ is unable to compute new group key.

D. GDH3

The GDH3 [13] consist of four stages. In first stage, it collects the contribution from members $M_i$. $M_{n-1}$ similar to GDH1. In the second stage, $M_{n-1}$ broadcasts $g^{x_1 \cdot x_{n-1}}$ to all members of group. In next stage, each member factors out its exponent $x_i$ from the receive value and sends the result to $M_{n-2}$. In the final stage, $M_n$ powers the received values from group members with its exponent $x_n$ and broadcasts the results to rest of members in the group. After receiving these values, each member can compute the group key.

Member $M_n$ has to save the content of original broadcast message from $M_{n+1}$ and response messages from other members (during stage2 and 3). In join event, member $M_n$ generates a new secret exponent $x_n$ and computes a new sub keys which will be forward to new member. New member $M_{n+1}$ computes a new group key $K_{n+1}$. It also raises received sub keys to its own secret exponent $x_{n+1}$ and broadcasts them to all members in the group. Now, the group members can perform the group key $K_{n+1}$.

Generation key in leave event is similar to GDH2.

E. Octopus Protocol

In this protocol [15], each large group split into 4 subgroups. Each subgroup performs its intermediary DH value $(I_{subgroup} = g^{x_1 \cdot x_{n/4}})$ and then subgroups exchange their intermediary DH value with each other. All group members are then able to generate the group key. The contribution from each subgroup members is collected by leader of each subgroup in order to perform intermediary DH value.

F. STR

Kim et al. has proposed STR [11], [16] protocol to extend Steer protocol to deal with dynamic group and network failure. They used a logical key hierarchy to minimize the number of key that held by group members.
STR key tree has two type of node, Leaf and Internal. Each member group is associated with a leaf node. An internal node \( IN_{\ell_{i,j}} \) always has two nodes, leaf node and another internal node \( IN_{\ell_{i,j}} \). The internal node \( IN_{\ell_{i,j}} \) is exception. It’s also a leaf node.

Each leaf node \( LN_{\ell_{i,j}} \) has a session secret key \( x_i \) which is kept secret by member \( M_i \). The blinded (public) key \( bx_i \) of each leaf node calculate as \( g^{x_i} \mod p \). Every internal node has secret key \( K_{\ell_{i,j}} \) and blinded key \( bk_{\ell_{i,j}} = g^{x_i} \mod p \). The internal node \( IN_{\ell_{i,j}} \), computes its secret key \( K_{\ell_{i,j}} \) by using of Diffie Hellman key agreement of its two children. The \( K_{\ell_{i,j}} \) is generated recursively as follow:

\[
K_{\ell_{i,j}} = (bk_{\ell_{i,j}}^{-1})^{x_i} \mod p = (bk_{\ell_{i,j}}^{-1})^{x_i} \mod p = g^{x_i} \mod p \text{ if } i > 1.
\]

For example in Fig. 1, \( K_1 = x_1, K_2 = g^{x_1} \mod p, K_3 = g^{x_2} \mod p \) and \( K_3 = g^{x_4} \mod p \).

In join event, the new user broadcast his join request to all current members. All members receive this request and determine if the former sibling of \( M_i \) joins to the group. If it’s, \( M_i \) is a leaf node directly below the \( M_i \). Otherwise \( M_i \) is a new root node.

If member \( M_d \) decides to leave the group, the other group members upon hearing theleave event begin to update its key tree by deleting the node \( LN_{\ell_{i,d}} \), corresponding to leaving node and its parent node \( IN_{\ell_{i,d}} \). The nodes above the leaving member \( M_d \) are renumbered and the sibling of \( M_d \) would be replaced by \( M_d \)'s parent. The sponsor \( M_s \) is a leaf node directly below the \( M_d \) (if \( d > 1 \)). Otherwise the sponsor is \( M_2 \). The sponsor selects a new secret key random and computes all key up to the root and broadcast to \( BT_{\ell_{i,d}} \). Now, all members can regenerate the new group key.

### G. TGDH

In Tree based Group Diffie Hellman (TGDH) [10], [17], all members maintain a virtual binary tree. The nodes in tree are denoted by \( <L, V> \), where \( 0 \leq V \leq 2^L - 1 \) since each level \( L \) has at most \( 2^L \) nodes. \( V \) indicates the sequential index of node at level \( L \) in tree. Each parent node \( <L, V> \) has two children, which are labeled as \( <L+1, 2V> \) and \( <L+1, 2V+1> \), respectively. The root of tree indicates by label \( <0, 0> \) as it’s located at level 0.

![Fig. 1. Example of STR key tree](image1)

In this protocol, all group members are hosted on leaf nodes respectively. Each node in tree except root is associated to a secret key \( K_{<L, V>} \), which is a Diffie Hellman private key, and a blinded (public) key \( BK_{<L, V>} = f(K_{<L, V>}) \), where \( f(x) = g^x \mod p \). The parent node computes its key \( K_{<L, V>} \) by using private keys of its two children, i.e. \( K_{<L, V>} = f(K_{<L+1, 2V>}, K_{<L+1, 2V+1>}) \). It means that each node’s secret key can be computed from secret key of one of its two children and blinded key of other child by using the Diffie Hellman key exchange protocol. All blinded key are broadcasted to all members of group.

Each member \( M_i \) at \( <L, V> \) knows the secret key \( K_{<L, V>} \) along path from itself to the root of tree. \( K_{<0,0>} \) at the root node is group secret key which is shared by all members in the group. Fig. 2 demonstrates a key tree with five members.

![Fig. 2. Notation of TGDH](image2)

When a new user is interested in joining to group, it broadcasts its request message that contains its blinded key. All pre-existing members receive this request and determine the join point in the tree and its sponsor. The sponsor is rightmost leaf in the subtree rooted at the insertion node. Each member updates the key by adding a new node for new member, new intermediate node and removing all secret key and blinded key from sponsor's leaf to the root. Then, sponsor node computes all secret key and blinded key along path from its leaf node to the root. Now, sponsor node broadcast all new blinded key to all members. All members upon receiving new blinded key would be able to generate the new root key.

If one of existing member \( M_i \) decides to leave the group, each member updates its key tree by deleting key corresponding to \( M_i \), the former sibling of \( M_i \) replace by \( M_i \)'s parent. The sponsor generate new secret key and computes all key and blinded key from itself to the root. After generating key, sponsor broadcasts new blinded keys \( BK \) to all members, which allows all members to compute the new group key. Sponsor at this case is the shallowest rightmost leaf of the subtree rooted at \( M_i \)'s sibling node.

### H. CRTDH

In Chinese Remainder Theorem and Diffie Hellman
(CRDTH) scheme, all members exchange their contributed key share by using the Diffie Hellman key exchange mechanism, and then the members independently but mutually generate the group key based on the chinese remainder theorem (CRT) [18].

Each node computes its Deffie Hellman public key \( y_i \) and broadcast it to all nodes. Next, Each node which receives others nodes public key, computes secret shared key with the sender of public key. After that each node selects a random key \( K \), which is less than all the received public keys in order to solve CRT value. The CRT value is broadcasted to other members. Each member calculates other members’ random key by using received CRT value. Every member now extracts other members’ random key from received CRT values, and then XOR them to make traffic key.

In join occurrence, one of closest member transmit hash value of traffic key as well as all group members’ Diffie Hellman public key to new member. New member computes DH and CRT and broadcasts new CRT to all group members. The pre-existing users can calculate the received CRT and extract the new member key \( K \). They XOR new \( K \) with current traffic key for providing new key.

In leave event from group, one of the remaining members selects a new key \( K \) and computes new CRT, and then broadcasts it to other members. Upon receiving new CRT, the new key \( K \) will be extracted from new CRT by other members. New traffic key will be obtained by XORing existing traffic key with new key \( K \).

IV. ANALYSIS AND COMPARISON

This section summarizes and compares cost of communication of the contributory key management protocols presented in section III during group operation: initial, join and leave.

The initial operation is the initial creation of the group key and organization of the key management infrastructure.

Join operation is adding a new member to the group during group communication and participate in present and future communication.

Leave operation is used to remove a member from group either voluntary or enforcing expulsion and won’t be able to decrypt next group communication messages.

Cost of communication contains number of rounds and number of exchange messages between members during group operation such as initial, join and leave. The exchange messages can include either unicast or broadcast messages.

We look at the number of round as a time unit that shows the number of steps taken in an operation.

The number of unicast message is sum of the messages every member sends to another single member in group.

The number of broadcast message is sum of all messages which is sent by a member to other members in a group for operation. This greatly affects the communication cost in compare with unicast.

The total number of message is sum of all unicast and broadcast messages which are sent in group during operation. This number is used to determine the total communication time.

Table I summarizes communication cost for the eight protocols which are investigated in section III. The numbers of current group member is denoted by \( n \). The height of key tree constructed by TGDH is also denoted by \( h \) that is equal\( \log_2 n \).

ING, GDH1 were originally designed to support only group formation. They cannot support join or leave operation in group. These protocols have to restart anew upon every changing in group membership. Octopus was designed to exchange key without using broadcast message. We don’t consider join and leave operation in this protocol.

From Table I, the number of rounds of all protocols except CRDTH in initial operation is relative to number of members \( n \) in a group. GDH1 in compare with other protocols has the biggest number of rounds with \( 2n-2 \). This means that GDH1 needs to take \( 2n-2 \) steps and consume more time to perform group key. The CRDTH has a fixed round number, which means that the number of interaction between members is independent from the number of group members.

In term of number of exchange messages, ING with \( n-1 \) messages in each round is the most expensive protocol. However, STR and TGDH exploit \( 2(n-1) \) broadcast messages. As every broadcast message is sent to all members of group, It makes sense that each broadcast message will consume more bandwidth than a given unicast message.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>No round Message</th>
<th>Unicast</th>
<th>Broadcast</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>ING [12]</td>
<td>Initial</td>
<td>n-1</td>
<td>n(n-1)</td>
<td>n(n-1)</td>
</tr>
<tr>
<td>GDH1 [13]</td>
<td>Initial</td>
<td>2(n-1)</td>
<td>2(n-1)</td>
<td>2(n-1)</td>
</tr>
<tr>
<td>GDH2 [13-14]</td>
<td>Initial</td>
<td>n</td>
<td>n-1</td>
<td>n</td>
</tr>
<tr>
<td></td>
<td>Join</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Leave</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>GDH3 [13]</td>
<td>Initial</td>
<td>n+1</td>
<td>2n-3</td>
<td>2n-1</td>
</tr>
<tr>
<td></td>
<td>Join</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Leave</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>Octopus [15]</td>
<td>Initial</td>
<td>2(n-4)/4 + 2</td>
<td>3n - 4</td>
<td>3n - 4</td>
</tr>
<tr>
<td>STR [11, 16]</td>
<td>Initial</td>
<td>n-1</td>
<td>2(n-1)</td>
<td>2(n-1)</td>
</tr>
<tr>
<td></td>
<td>Join</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Leave</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>TGDH [10, 17]</td>
<td>Initial</td>
<td>n</td>
<td>2(n-1)</td>
<td>2(n-1)</td>
</tr>
<tr>
<td></td>
<td>Join</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>Leave</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CRDTH [18]</td>
<td>Initial</td>
<td>2</td>
<td>2n-2</td>
<td>2n</td>
</tr>
<tr>
<td></td>
<td>Join</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Leave</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

In join operation, all protocols require two rounds. All of them use a constant number of messages, which are independent from number of members in the group. TGDH and STR employ three broadcast messages that in contrast to other protocols probably consume more bandwidth.

All protocols have the same communication cost in leave event. They have one round consisting one message.

V. CONCLUSION

This paper has presented a review of group key management, particularly contributory schemes which
employ public key exchange algorithm. It looked at the initial operation when a number of members of a group are going to generate a shared traffic key. Meanwhile, the used method for provision of backward secrecy after joining a new member to the group as well as forward secrecy after leaving a pre-existing member from the group have been investigated. It also provided a qualitative evaluation of the presented approach. The investigated schemes then summarized and analyzed in term of communication cost. In the initial operation, communication requirement of all contributory schemes (except CRTDH) are raised by increasing number of members. Otherwise, they are independent from number of members in join or leave operation.

REFERENCES


Babak Daghighi received his B.Sc. in Computer Science from Azad University (Tehran branch), Iran, and M.Sc. from University of Malaya (UM), Malaysia. He is now continuing his Ph.D studies in Computer Science in University of Malaya. His research interests include cryptography, network security, and wireless system with an emphasis on group key management in wired and wireless networks.

Laiha Mat Kiah received her B.Sc. (Hons) in Computer Science from University of Malaya (UM), Malaysia in 1997, M.Sc. in 1998 and Ph.D. in 2007 from Royal Holloway, University of London UK. She is currently the Deputy Dean of Faculty of Computer Science and Information Technology, University of Malaya. Her current research interests include wireless/mobile/cloud security, secure group communication and key management. She is also interested in routing protocols, ad-hoc networks and secure algorithms.

Saeedeh Afkhami Rad received her B.Sc. in Computer Science from Azad University ( Mashad branch), Iran. She is now continuing her master studies in Computer Science in Shahid Beheshti University, Iran. Her research interests include database.