Group Key Management on Tree-based Braid Groups

Thanongsak Aneksrup [1], Pipat Hiranvanichakorn [1]

[1] National Institute of Development Administration, School of Applied Statistics
(tnongs@yahoo.com, pipat@as.nida.ac.th)

Abstract

The characteristic of mobile ad hoc networks (MANETs) build temporary dynamic network without any fixed infrastructure and limitation of power and computing capability present the challenge in secure communication. The existing key management protocols in wire network still do not suitable in MANETs. In this paper, we propose the secure and efficient contributory group key agreement protocols. The protocols are based on braid groups cryptographic with key tree by avoiding modular exponential operation in Diffie-Hellman protocol. The braid groups and key tree are two techniques what can minimize communication and computing cost of generating group key. The braid groups only use product and inverse operation but sufficient complexity, since the hardness of the generalized conjugacy search problem is applied in our protocol. The dynamic operation protocol, especially join and merge protocol, use maximum signal strength between current member so call *"director"* and new member in order to achieve minimum hop communication, shortest range and fastest transfer rate. The director of leave and partition protocol is the member that has maximum number of one-hop neighbors for fastest broadcasting the information form director to every members. The constant round in communication of each protocol is designed with computation cost in serial number of braid permutations as O(log *n*). Our approach is simple, secure and efficiency for group key management in mobile ad hoc networks.

1. Introduction

A whole new generation of mobile devices, such as cellular phone, personal digital assistant (PDA), computer laptop, etc., are commercially available. These devices capable communicate both wired and wireless network. The special type of wireless network forming temporary dynamic network, without the aid of any fixed infrastructure is mobile ad-hoc network that implies an absence of a trust entity for manage the security in networks, such as router, CA and servers. The security is more challenge than wired or wireless networks. The existing security solutions applied in traditional network with a static configuration may not suitable directly for preventing member nodes according to nature of mobile ad-hoc network that causes frequently changed network topology. Many applications of mobile ad-hoc networks involve a large number of nodes, for example is mission of fire fighters in rescue task or of soldier during battle. Therefore it is necessary to provide support for secure group communication.

The common share group key is necessary for secure group communications. There are three group key agreement schemes including centralized, distributed and contributory. First scheme, the centralized group key agreement, where a node, central key server, perform generating and distributing group key is simple but is not suitable for dynamic network topology since server must be anywhere and anytime for managing every member nodes in the group. The second scheme, distributed group key agreement is more appropriate to group communication, especially over unreliable networks. The key server is dynamically selected from current group members. This approach has drawback, it requires the key server to maintain long-term pairwise secure channels with all current member in order to distribute share group key. Last scheme, contributory group key agreement demands each member to contribute for the group key generation. This scheme is fault tolerant and avoids the problem with the centralize trust and single point of failure. The first key exchange protocol was proposed by Diffie and Hellman [2] what based on the difficulty of the decision Diffie-Hellman over finite fields. Most group key agreement protocols extended the Diffie-Hellman key exchange protocol in multiparty instants. Some new protocol adapted the Diffie-Hellman key exchange protocol over braid groups [1, 3, 4, 5, 6]. Ko et al. [5] studied the key exchange protocol on braid groups that based on the Diffie-Hellman version of conjugacy

problem, so call Ko-Lee problem. Their proposal changed the concept on number theory that widely use in cryptographic to braid group. Lee at el. [6] extend two party key agreement from [5] to be the group key agreement on braid group based on the hardness Ko-Lee problem and Cliques. They extended protocol to authenticated group key agreement. Kui at al. [1] design the group key agreement based on braid group and Diffie-Hellman key exchanges protocol with dynamic operation protocol including join, leave, merge, partition and refresh protocols. The Diffie-Hellman key agreemetn is partially contributory scheme that there are many drawbacks, such as requiring sequencing among the group member, and the highest indexed member (group controller) presents a single point of vulnerability. Therefore the protocol based on Diffie-Hellman key agreement is not suitable in environment of mobile ad hoc network.

Our proposal, we design the contributory group key agreement on group communication of mobile ad-hoc networks. We combine two important techniques in group key management: 1) braid groups key exchange to avoid modular exponential operation as well as achieve secure and fully distributed protocol and 2) key-trees to efficiently generate group key that the number of rounds to construct the group key can be reduced to the logarithm of the number of members as well as it can reduce the communication, computation and storage overhead. The communication reduces to constant round that is not depends on number of members. The key tree is nearly physical since in protocol new member needing to join the group will detect maximum signal strength of current member and transfer information with that node before generating share key.

The remained section is organized as follows. Section 2 mention the background of braid groups and key exchange based on braid groups. Section 3 describes the tree-base braid groups protocol for group key management including membership operation, join, leave, merge, partition and key refreshing. Section 4 analyzes the security of our protocol. Section 5 analyzes the performance of our protocol compare with others in communication and computation cost.

2. A braid groups cryptographic

2.1 Preliminaries of Braid Group

The braid groups were first systematically proposed by Emil Artin. He introduced the Artin generators $\sigma_1, \sigma_2, \ldots, \sigma_{n-1}$ for the n strand braid groups what is denoted as B_n. The integer n is called the *braid index* and each element of B_n is called an n-braid. The B_n is a collection of disjoint n strings. A general n-braid is constructed by iteratively applying the σ_i ($i = 1,.., n-1$) operator, which switches the lower endpoints of the i^{th} and $(i+1)^{th}$ strings keeping the upper endpoints fixed with the $(i+1)^{th}$ string brought above the i^{th} string. If the $(i+1)^{th}$ string passes below the i^{th} string, it is denoted as σ_i^{-1}. Any n-braid can be expressed as a *braid word*, e.g., $\sigma_3\sigma_2\sigma_1^{-1}\sigma_1^{-1}$ is a braid word, a in Fig. 1, in the braid group B_4. The inverse of braid word is constructed by reversing each crossing sequentially. For example is shown in Fig.1, $b = \sigma_1^{-1}\sigma_3^{-1}\sigma_2^{-1}$ and $b^{-1} = \sigma_2\sigma_3\sigma_2$ and The multiplication of two braids word, ab, is the braid achieved by positioning b on the bottom of a. The identity is braid is not intertwining strings. The definition of braid groups is shown in Fig. 1:

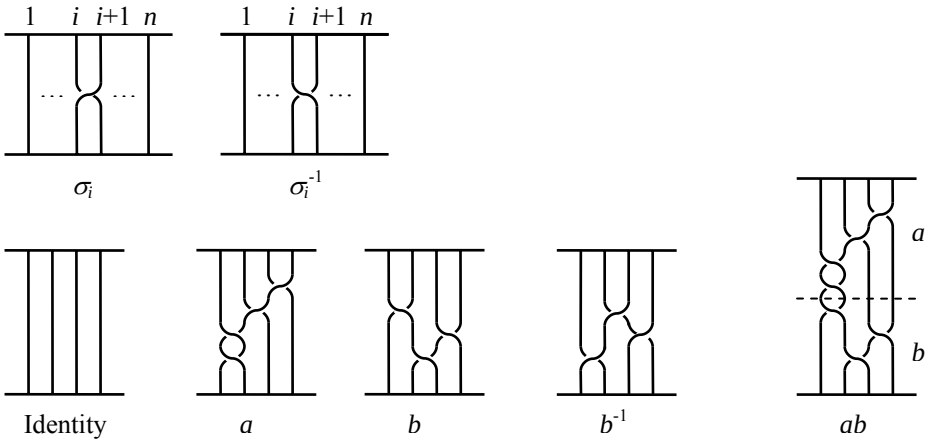

Fig. 1 Definition of braid groups

The relation of n-braid groups B_n are as follows and shown in Fig. 2:

(1) $\sigma_i \sigma_j = \sigma_j \sigma_i$ where $| i - j | \geq 2$

(2) $\sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1}$

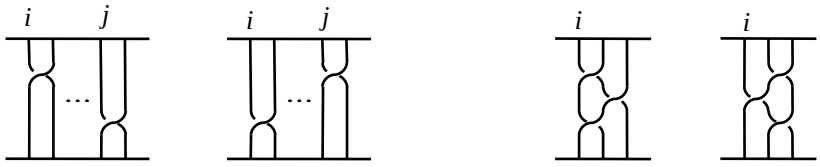

Fig. 2 The relation of braid groups

2.2 Hard problem in the braid groups

We explain braid groups in generalized conjugacy search problem [7] that is applied in our protocol in order to increasing strength of key. We say that x and y are conjugate if there is element a such that $y = a\,x\,a^{-1}$ for $m < n$, B_m can be considered as a subgroup of B_n generated by $\sigma_1, \sigma_2, \ldots, \sigma_{n-1}$.

Instance: $(x, y) \in B_n \times B_n$ such that $y = a\,x\,a^{-1}$ for some $a \in B_n$, $m \le n$.
Objective: Find $b \in B_m$ such that $y = b\,x\,b^{-1}$.

We consider two subgroups B_l and B_r of B_{l+r}. The B_l and B_r are made by braiding left l strand and right r strand among $l+r$ strand respectively. The cumulative property for any $a \in B_l$ and $b \in B_r$ is $ab = ba$. The adequately complicated $(l+r)$-braid is selected as $x \in B_{l+r}$. Thus the one-way function is shown as follow:

$$f : B_l \times B_{l+r} \rightarrow B_{l+r} \times B_{l+r}, \quad f(a,x) = (a\,x\,a^{-1}, x) \tag{1}$$

The function is simply to calculate $a\,x\,a^{-1}$ for given a and x but need exponential time to compute a from the information. This one-way function is based on the generalized conjugacy search problem.

2.3 Key exchange on braid groups

In Crypto 2000, Ko et al. proposed a new public-key cryptosystem using braid groups based on hard problem of the conjugacy problem [5]. They proposed new key agreement using braid groups which is a Diffie-Hellman version. The key agreement protocol is shown as follow:

(1) ***Preparation step*** : Assume Alice and Bob want to securely communicate on public channel. It mean Alice and Bob have to share a secrete key. An appropriate pair of integer (l, r) and a sufficiently complicated $(l+r)$-braid $\alpha \in B_{l+r}$ are selected and published.

(2) ***Key agreement protocol*** :
(a) Alice selects a random secrete braid $a \in B_l$ and sends $y_1 = a\alpha\,a^{-1}$ to Bob.
(b) Bob selects a random secrete braid $b \in B_r$ and sends $y_2 = b\alpha\,b^{-1}$ to Alice.
(c) Alice receive y_2 and computes the share key $K = a\,y_2\,a^{-1}$.
(d) Bob receive y_1 and computes the share key $K = b\,y_1\,b^{-1}$.

Since $a \in B_l$ and $b \in B_r$, $ab = ba$, yield

$$K = a\,y_2\,a^{-1} = a\,(b\alpha\,b^{-1})\,a^{-1} = b\,(a\alpha\,a^{-1})\,b^{-1} = b\,y_1\,b^{-1}$$

Thus Alice and Bob yield the same secrete key K.

2.4 Multiparty key agreement on braid groups

In 2002, H. K. Lee et al. extended key exchange protocol in section 2.3 to a group key agreement protocol based on Cliques [6]. In protocol, it consider n subgroups $B_{l1}, B_{l2}, \ldots, B_{ln}$ of l-braid group B_l where $l = l_1+l_2+\ldots+l_n$. B_l consisting of braids made by braiding l_i -strands from the left among l-strands with the order $l_1, l_2, \ldots, l_n$. For any $r_m \in B_{lm}$ and $r_n \in B_{ln}$ with $m \neq n$, $r_m r_n = r_n r_m$. The $\alpha \in B_l$ be a sufficiently complicated l-braid are selected and published. Supposing n members need to share a key. The protocol is shown as follow steps:

Round i, $i \in [1, n-1]$:

M_i selects a random $r_i \in B_{li}$,

$M_i \rightarrow M_{i+1}$: $\{ r_i \ldots \hat{r}_j \ldots r_1 \alpha\, r_1^{-1} \ldots \hat{r}_j^{-1} \ldots r_i^{-1} \mid j = 1, 2, \ldots, i \}$ and $r_i \ldots r_j \ldots r_1 \alpha\, r_1^{-1} \ldots r_j^{-1} \ldots r_i^{-1}$

where $\hat{r}_j$ means that r_j does not exist.

Round n:

M_n selects a random $r_n \in B_{ln}$,

$M_n \rightarrow M_i$ $i \in [1, n-1]$: $\{ r_n \ldots \hat{r}_i \ldots r_1 \alpha\, r_1^{-1} \ldots \hat{r}_i^{-1} \ldots r_n^{-1} \}$.

The group key is achieved as $r_n \ldots r_1 \alpha\, r_1^{-1} \ldots r_n^{-1}$

Their protocol is partially contributory model since each operation has to select group controller (highest index) that it will generate share key and distribute to other members, group controller is risk entity to present a single point of vulnerability. This protocol have drawbacks, they are slow computation O(n) each operation. Also communication overhead is expensive.

3. Tree-based braid groups protocol

There are many protocols to propose for group key agreement in ad hoc network that they based on Diffie-Hellman key exchange protocol. These protocols are designed to reduce the number of communication rounds. Otherwise, there some protocol that based on braid groups. Our protocol is designed based on braid groups. In this paper the authentication is not determined in our group key agreement protocol. All communication is public but authentic. There are two importance techniques including key tree and braid groups key exchange to design the protocol in considering limited computing, storage and power capacities in ad hoc network. We describe these techniques in following section. Our technique based on generalized conjugacy search problem that mention above.

3.1 Key tree

Key tree is earliest proposed by Wallner et al. [8] and adapted in contributory key agreement by Y. Kim et al. [7]. The tree structure is widely used to reduce the communication, computation and storage overhead. The number of rounds to form the group key can reduce to the logarithm of the group size. We also applied this technique in our protocol since it is best and suitable technique to use for contributory group key agreement in mobile ad hoc network according to above reason. We describe the notation and definition of key tree that is applied in our protocol in following. The sample of key tree is shown in Fig 3. The binary tree, every node is either a leaf or a parent of two nodes, is used in key tree. The nodes are denoted $[h, v]$. Each node $[h, v]$ is associated with the secret key $K_{[h, v]}$ and the blinded key $BK_{[h, v]}$ equal to $f(K_{[h, v]})$ where function $f(\ )$ is based on braid groups key exchange that we describe in following section. The member is located at the leaf node. The information of intermediate nodes, key and blinded key, computes from the information of two children node to achieve the subgroup key. The *key-path* is referred that the member M_i, where $1 \le i \le n$, at the leaf node $[h, v]$ knows every key along path from $[h, v]$ to root node. The *co-path* is the set of siblings of each node in the key-path of member M_i. The group secrete key is key at the root node $K_{[0, 0]}$ that can compute from all blind keys on the co-path and its session random $K_{[h, v]}$ at M_i view.

For example, in Fig. 3, M_3 knows every key in node at position {[3,3], [2,1], [1,0], [0,0]} as key-path that member node can compute the intermediate node key on key path from key of one child node and the blinded key of other child node, and every blinded key in node at position {[3,2], [2,0], [1,1]} as co-path that receive from broadcasting of leader, so called "*director*", in any membership event.

3.2 Braid groups key exchange

We suppose n subgroups (members) $B_{g1}, B_{g2}, \ldots, B_{gn}$ of g-braid groups B_g where $g = g_1 + g_2 + \ldots + g_n$. B_g consisting of braids made by braiding g_i -strands from the left among g-strands with the order $g_1, g_2, \ldots, g_n$. For any $s_m \in B_{gm}$ and $s_n \in B_{gn}$ with $m \neq n$, $s_m s_n = s_n s_m$. The $\beta_{in} \in B_q$, where $B_q \subseteq B_g$, be a sufficiently complicated braid are selected and published. The public braid word, β_{in}, at intermediate node including root node is selected from the union set of member's braids subgroups under that intermediate node. Supposing n members need to share a key. Each member selects the secret key. The blinded key BK is generated by $f(K)$ that is equal $K\beta_{in}K^{-1}$. Every key $K_{[h, v]}$ is computed

recursively as follows:

$$\begin{aligned} K_{[h,v]} &= K_{[h+1,2v]}\, BK_{[h+1,2v+1]}\, K^{-1}{}_{[h+1,2v]} \\ &= K_{[h+1,2v+1]}\, BK_{[h+1,2v]}\, K^{-1}{}_{[h+1,2v+1]} \\ &= K_{[h+1,2v]}\, K_{[h+1,2v+1]}\, \beta_{[h,v]}\, K^{-1}{}_{[h+1,2v]}\, K^{-1}{}_{[h+1,2v+1]} \\ &= K_{[h+1,2v+1]}\, K_{[h+1,2v]}\, \beta_{[h,v]}\, K^{-1}{}_{[h+1,2v+1]}\, K^{-1}{}_{[h+1,2v]} \end{aligned} \qquad (2)$$

where $K_{[h+1,2v]} \in B_{[h+1,2v]}$ and $K_{[h+1,2v+1]} \in B_{[h+1,2v+1]}$ with $B_{[h+1,2v]} \neq B_{[h+1,2v+1]}$, thus $K_{[h+1,2v]}K_{[h+1,2v+1]} = K_{[h+1,2v+1]}\, K_{[h+1,2v]}$.

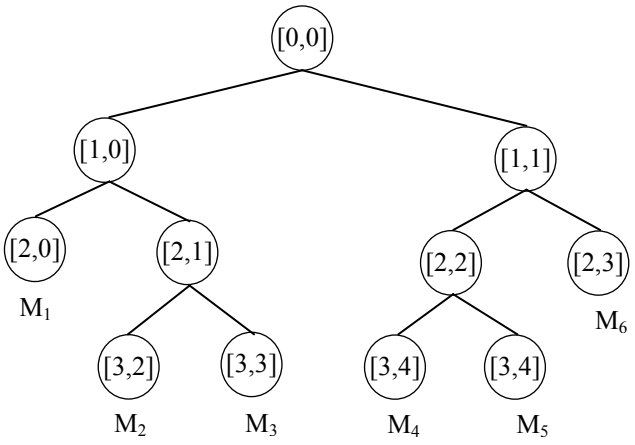

Fig. 3 Notation of tree

The key generating at [*h*, *v*] requires the information compose of one key and one blinded key of two child nodes. The root key at [0, 0] is group secret key that is shared by all current members. A group key can be computed from one member's secret key of leaf node value for this node view and all blind keys on the co-path to the root.

We show in example that all member nodes achieve the same group key in contributory manner. We label leaf node as A, B and C for ease to understand that shown in Fig.4. Assume each leaf node (member node) select own random secrete braid, A select a $\in B_a$, B select b$\in B_b$ and C select c $\in B_c$.

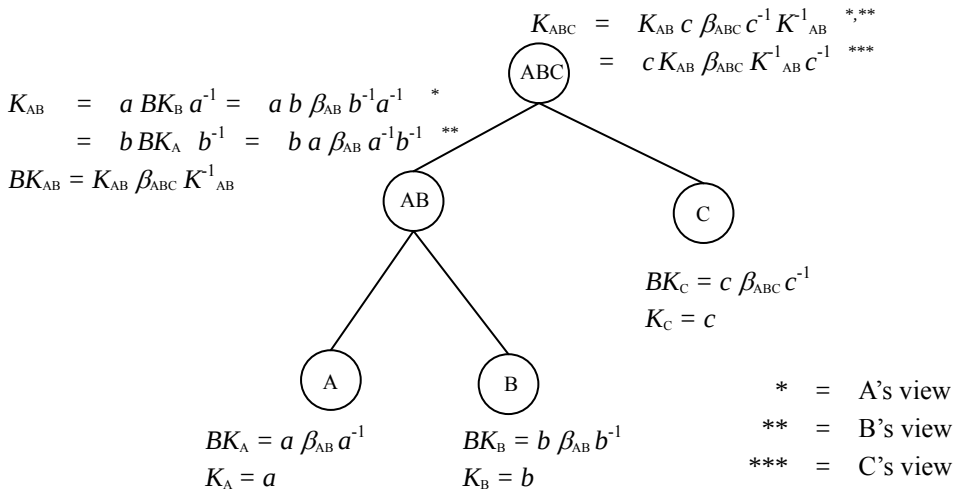

Fig. 4 Group Key Generating

Each member node generates the group key K_{ABC} in contributory manner by equation (2) to achieve as follow:

A's view : $K_{ABC} = a\, b\, \beta_{AB}\, b^{-1}\, a^{-1}\, c\, \beta_{ABC}\, c^{-1}\, a\, b\, \beta^{-1}{}_{AB}\, b^{-1}\, a^{-1}$

B's view : K_{ABC} = $b\ a\ \beta_{AB}\ a^{-1}\ b^{-1}\ c\ \beta_{ABC}\ c^{-1}\ b\ a\ \beta^{-1}{}_{AB}\ a^{-1}\ b^{-1}$

C's view : K_{ABC} = $c\ a\ b\ \beta_{AB}\ b^{-1}\ a^{-1}\ \beta_{ABC}\ a\ b\ \beta^{-1}{}_{AB}\ b^{-1}\ a^{-1}\ c^{-1}$ = $c\ b\ a\ \beta_{AB}\ a^{-1}\ b^{-1}\ \beta_{ABC}\ b\ a\ \beta^{-1}{}_{AB}\ a^{-1}\ b^{-1}c^{-1}$

The braid sequences of root key at each node view shown as Fig.5 are equal in each subgroup to imply as same braid. We conclude that braid group can be applied in key tree. Therefore root key that generated by each member node can be session group key.

3.3 Group Key Management on Tree-based Braid Groups (TBG)

Our key tree scheme based on [7] that each node can compute intermediate key from own secret key and sibling blinded key of the co-path node. Therefore the member node at leaf knows all keys on the key-path. This instance shows that the member is not necessary to know all blinded keys for generating the group key but knowing the all blinded key in our protocol in each member is provided for membership change to be more efficient and robust.

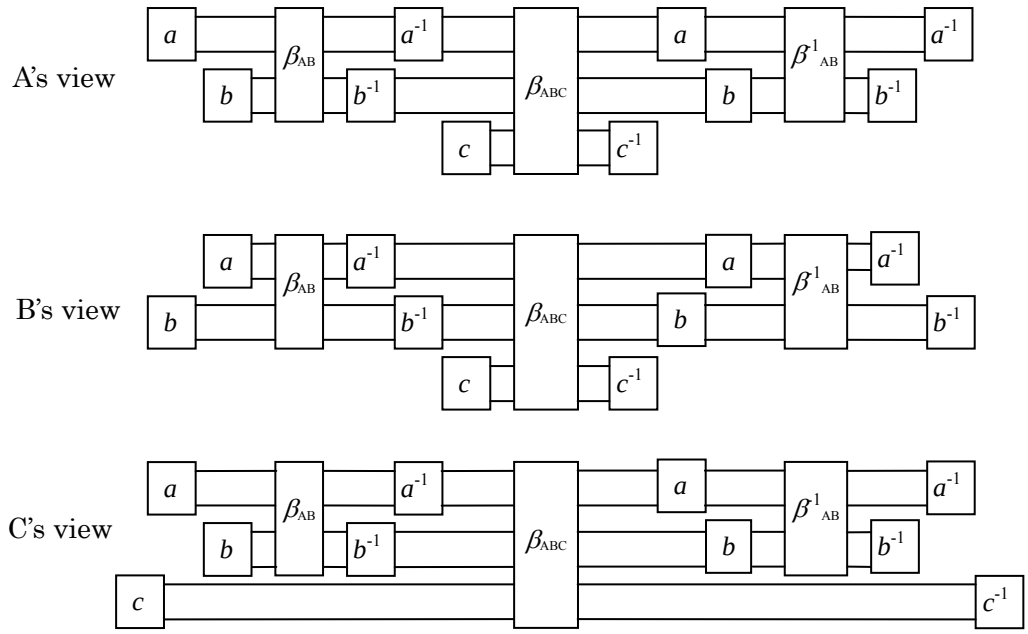

Fig. 5 Braid Permutation Sequence of each Perspective

The most of past research based on logical key tree. This scheme may be multi-hop communication between new member and leader of current member. In other words, some instance new member may communicate with leader that is longest range to compare with other current member. The new idea in our protocol is physical tree that the new member tries to communicate with leader at shortest range. We use maximum signal strength between new member and leader that in this paper call **"director"**. The signal strength achieve from embedded hardware in mobile device such as 802.11b or WiFi. The technique will reduce communication time and transfer information of new member to director as fast as possible.

The following section, we describe the protocol that construct the group key management on tree-base braid groups (TBG) including the following operations:

Join: a new member requests to add the group
Leave: a current node requests to remove the group
Merge: a group requests to merge the current group
Partition: a subset of member request to split from the current group
Key refreshing : a current member request periodically refreshed

The new key tree containing all blinded key, number of one-hop neighbors in group, number of current group members, number of merging group members are denote as $T^{*}[BK]$, N_{oh}, n and n_m, respectively.

3.4 Join Protocol

The group has n members, $\{M_1,..,Mn\}$. The new member M_{n+1} need to join the group by discovery the maximum signal strength with the current node in order to transfer the information in minimum hop and shortest range, this node is director in session. Later the new member send JOIN request message that contains its own blinded key to director. The insertion point of new member to key tree is director node. The director creates a new intermediate node and a new member node, and promotes the new intermediate node to be the parent node of itself and new member node. Next, the director selects new session random key and computes keys and blinded keys going up to the root. The director broadcasts the new key tree which contains only blinded keys to all other members. All other members update their key tree and compute the new group key. Fig. 6 shows an example of M_6 joining a group where director as M_4. This instance, it means that the M_6 is nearest with M_4. The conclusion of join protocol is shown as following:

Step 1: The new member discovers the maximum signal strength of current member (director) and sends request to join and its blinded session random key

$$M_{n+1} \xrightarrow{[BK_{n+1}]} director$$

Step 2: The director node updates its session random key, updates key tree, computes keys and blinded keys, and broadcast the new key tree containing the only all blinded key.

$$director \xrightarrow{T^*[BK]} M_i\ ,\ i \in [1,n+1]$$

Step 3: Each member computes the group key

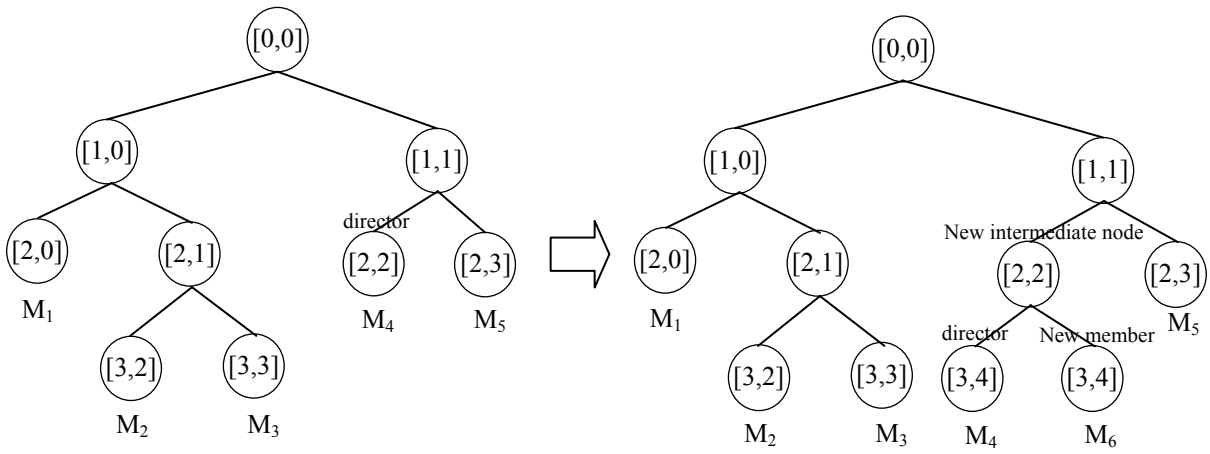

Fig. 6 Tree update: M_6 join, M_4 as director

3.5 Leave Protocol

We begin with n current member and the member M_r need to leaves the group. In this instance the director is a member that has maximum number of one-hop neighbors. The challenge members are particularly, every member of subtree root in side of leaving member location excluding M_r, because director only calculation the new blinded keys that subtree root, another subtree root is not necessary to update blinded keys. First, when the group detects that M_r leave the group, the members of subtree root in side of M_r location broadcast the number of one-hop neighbors, N_{oh}, for selecting the director. The member who has maximum number of one-hop neighbors is director of leave event. The reason of using maximum number of one-hop neighbors to select the director is the most of members receiving message from director in one-hop transmission that is fastest broadcasting message process to all destination members. After that, the director updates key tree by deleting the leaf node of M_r. The M_r's parent node is instead of director node. The director chooses the new session random key and computes keys and blinded keys going up to the root. Next, the director broadcasts the new key tree containing only blinded keys to all other member. The remainder members compute the new group key. Fig. 7 shows an example of M_3 leaving a group

where director as M_1 that means the M_1 has maximum number of one-hop neighbors. The conclusion of leave protocol is shown as following:

Step 1: Every remaining node of subtree root in side of leaving member location broadcast the number of one-hop neighbor.

$$M_i - M_r,\ i \in [1,n] \xrightarrow{N_{oh}} M_i - M_r,\ i \in [1,n]$$

Step 2: The director is node that has maximum number of one-hop neighbor. The director update the key tree, chooses the new session random key, compute keys and blinded keys and broadcast the new key tree.

$$director \xrightarrow{T^*[BK]} M_i - M_r,\ i \in [1,n]$$

Step 3: Each member computes the group key

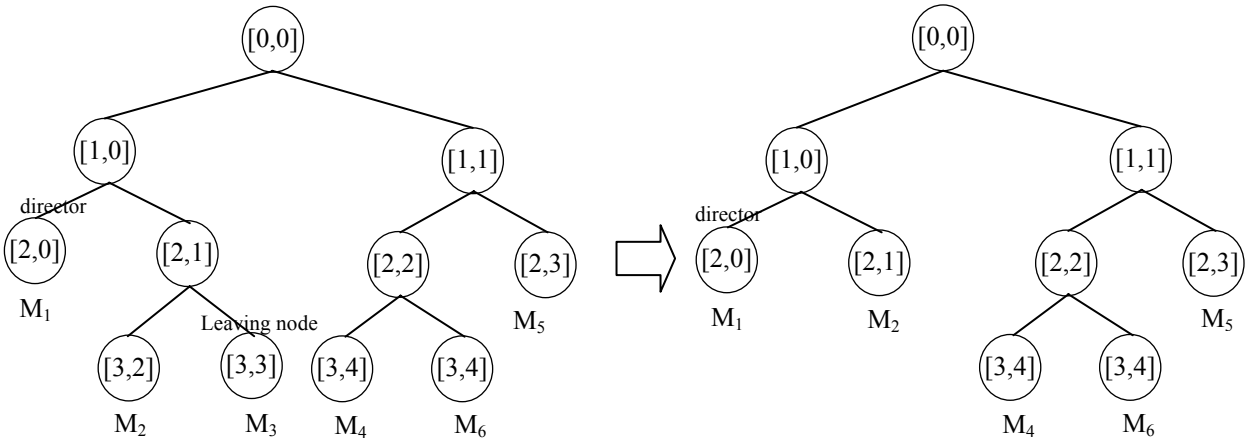

Fig. 7 Tree update: M_3 leave, M_2 as director

3.6 Merge Protocol

We assume that m merging group members require to merge with n current group members. Each member in merging group detects and achieves maximum signal strength what is measured as closest members between itself and current group member. The members in merging group challenge to find maximum signal by broadcasting the maximum signal strength. The member that has maximum of maximum signal strength is director of merging group. The current group director is member that has maximum signal strength with merging group director. The broadcasting of members increases the number of messages but not reduce the efficiency, because the broadcasting process in concurrent. Otherwise, using the maximum signal strength helps the transmission of two directors to be faster, since both directors are shortest range. Thus the communication cost in challenge to find director is not impact totally communication cost. The insertion node is director of current group. For efficiency of protocol, the merging group tree structure is not changed anything. Simply, root node of merging group is child node of new intermediate node in current group. The new intermediate node is generated by current group director at position of director node which its child nodes are director and a merging group tree. The merging group director chooses new session random key, computes blinded key and sends MERGE message with merging group key tree containing all blinded key to current group director. After the current group director updates key tree by combining the merging group key with above condition, the director refreshes session random key, computes keys and blinded keys, and broadcasts the key tree containing the all blinded keys to all members in new group. Finally, the group key is calculated independently by each member. Fig. 8 shows an example of merge operation. The members that have maximum of maximum signal strength are M_1 and M_8. Then the member M_8 is merging group director and the member M_1 is current group director. The merging point is [2,0] node, also this node is new intermediate node. The conclusion of merge protocol show as following:

Step 1: The members in merging group M_m challenge the maximum signal strength between merging member and current member in current group M_n. The signal strength between member in merging group and current group is denotes as $S_{m,n}$.

$$M_{mj}, j \in [1, m] \xrightarrow{\text{Max}(S_{m,n})} M_{mk}, k \in [1, m]$$

Step 2: The maximum of maximum signal strength is the director of merging group that chooses new session random, compute blinded key and send update tree to director of current group.

$$M_m \textit{ director} \xrightarrow{T^*_m [BK]} M_n \textit{ director}$$

Step 3: The director node updates its session random key, updates key tree, computes the all blinded key, generates the group key and broadcast the new key tree containing the only all blinded key.

$$M_n \textit{ director} \xrightarrow{T^*_{n+m}[BK]} M_i, i \in [1,n+m]$$

Step 4: Each member computes the group key

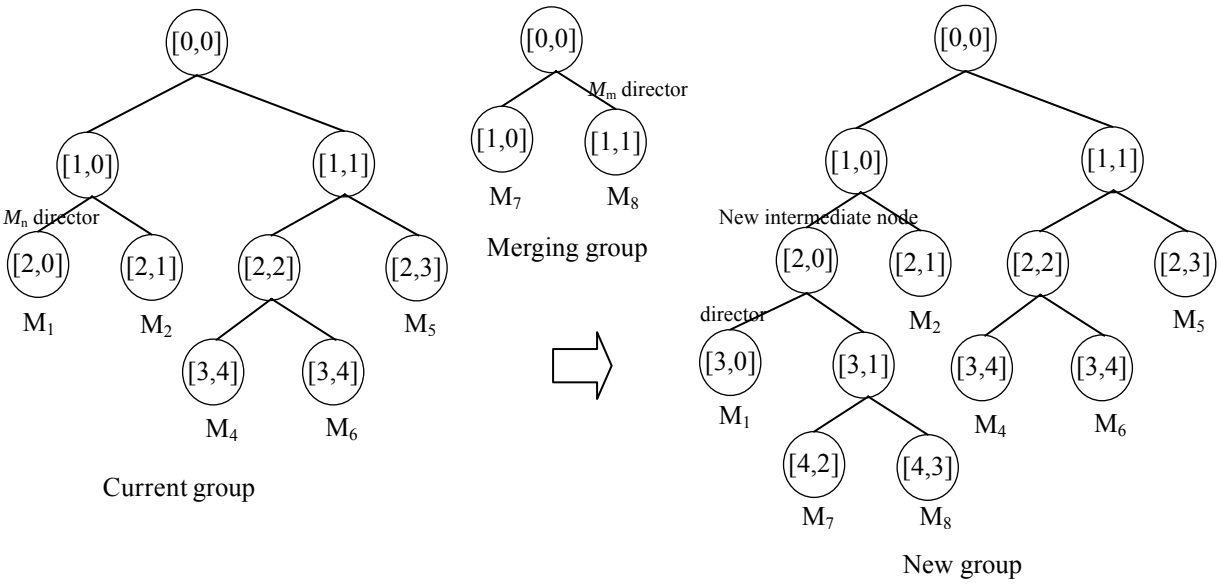

Fig. 8 Tree update: Merge Protocol

3.7 Partition Protocol

The partition operation occur when a network fault. The partition protocol is actually presents a concurrent multiple leave from group. The multiple members M_p need to leaves the group. The remaining members challenge to be director after knowing the occurring of partition event. The concept of partition protocol is quite similar leave protocol, but the challenge members to be director are every remaining member since the leaving nodes maybe exist both subtree root. In the first round of protocol, each remainder broadcasts the number of one-hop neighbors in group, N_{oh}. Each member compares the number of one-hop neighbors, when obtain from all remaining member. The member who has maximum number of one-hop neighbors is selected as director for partition operation. The reason of using number of one-hop neighbors to select the director is same as leave protocol that we mentioned above. As the leave protocol, after the director delete all leaving member in key tree chooses new session random key, it computes keys and blinded key going up to the root, and broadcasts the key tree with blinded keys to reminder member. Finally, each member computes the new group key. Fig. 9 shows an example of partition operation when M_1 and M_4 leave, and M_6 as director that means the M_6 has maximum number of one-hop neighbors. The conclusion of partition protocol show as following:

Step 1: Every remaining node broadcast average signal strength.

$$M_i - M_p\ ,\ i \in [1,n] \xrightarrow{N_{oh}} M_i - M_p\ ,\ i \in [1,n]$$

Step 2: The director is node that has maximum number of one-hop neighbor. The director update the key tree, chooses the new session random key, compute keys and blinded keys and broadcast the new key tree.

$$director \xrightarrow{T^*[BK]} M_i - M_p\ ,\ i \in [1,n]$$

Step 3: Each member computes the group key

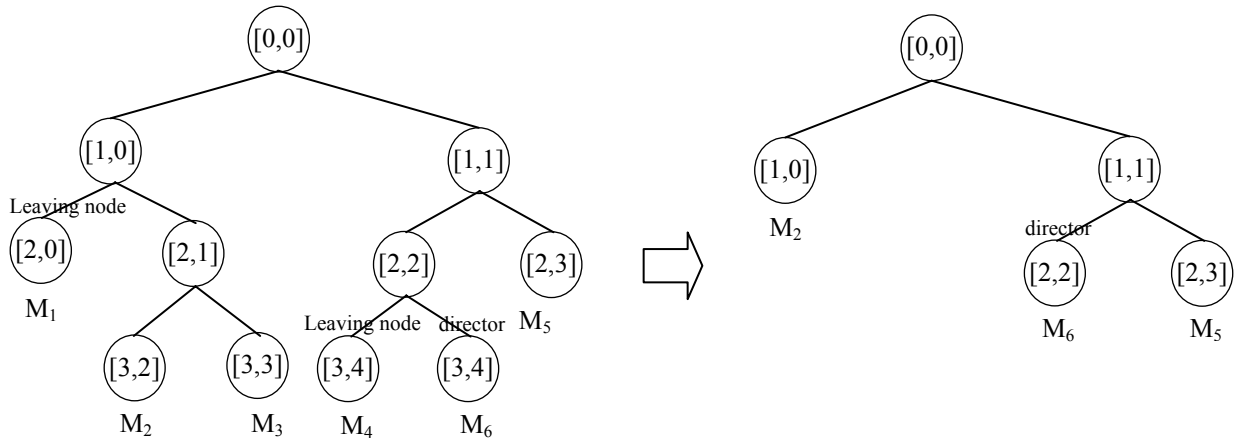

Fig. 9 Tree update: Partition Protocol

3.8 Key Refreshing

Key refreshing is MANETs is necessary, since most nodes can be easily compromised due to their mobility and physical vulnerability. Then the key refresh should be occur periodically in order to limiting exposure due the loss of keys and limiting the amount of ciphertext available to cryptanalysis for given group key. In protocol the node that needs to refresh the key is act as director. In similar way of other protocol, the director chooses the new session random key, compute keys and blinded keys up to the root, and broadcast updating key tree. All members compute the new group key. The conclusion of key refreshing protocol show as following:

Step 1: The director (refreshing node) chooses new session random key, computes keys and blinded key and broadcasts new key tree containing blinded key.

$$director \xrightarrow{T^*[BK]} M_i\ ,\ i \in [1,n]$$

Step 2: Every member computes the group key

4. Security

This section, we show that TBG satisfies forward and backward secrecy. It also implies to satisfy as key independence. The passive adversaries are unable to compute future and previous group key although they know all previous key tree and new key tree respectively, since the director refreshes the session random key every event.

First, we consider the forward secrecy, note that members that leave the group or passive adversaries who know a contiguous subset of old group key are unable to compute future group key. The forward secrecy is determined in leave and partition event. Assume *A* as leaving member at position *a* in key tree T. *A* knows all secret keys on key-path that are valid during its group membership. However the director of the leave and partition event update own session random key s_d and causes the change of all keys and blinded keys in key-path. Therefore *A* is unable to compute the subsequent group key, because the key tree information is changed. Thus TBG protocol provides the forward secrecy.

Later, we consider the backward secrecy to show that new group members are unable to compute old group keys. Assume *A* becomes a new member at position *a* in key tree T. As a new member *A* is able to compute all keys on key-path. The director of the join and merge event update own session random key s_d and causes the change of all keys and blinded keys in key-path. Therefore *A* is unable to compute previously used group key, since *A* can only compute changed keys due to changed key tree information. Thus TBG protocol satisfies the backward secrecy.

The combination of forward and backward secrecy we follow that TBG protocol satisfies key independence.

5. Performance

This section analyzes the communication and computation cost for join, leave, merge and partition protocol of TGB. We compare both cost with TGDH [7], braid groups based on Diffie-Hellman key agreement [5] and our proposed protocol TBG. We is analyzed the communication and computation costs for join, leave, merge, and partition protocol. The number of rounds, the total number of messages, the serial number of exponentiations, and serial number of braid permutations. Table 1 and Table 2 summarize the communication and computation cost respectively in three protocols. The number of current members, number of merging members, and then number of leaving members are denoted by *n*, *m*, and *p* respectively.

Firstly, we determine the communication cost is shown in Table 1. The number of rounds on TBG is constant in every operation but the other protocols are not constant on any event. The number of rounds on partition operation in TGDH depends on height of key tree, merge operation in Braid groups on IKA.2 depend on number of merging members, but every operation in TBG does not depend on number of members that dynamic movement. The number of rounds in TBG is equals to TGDH and Braid groups on IKA.2 in join protocol. The number of rounds in TBG are more than both TGDH and Braid groups on IKA.2 at one round, since the remaining members in TBG have to broadcasting the challenge message including the number of one-hop neighbors in first round. In merge protocol, the number of rounds in TBG is more than TGDH at one round, but less than Braid groups on IKA.2 about *m* round. In partition protocol, the number of rounds in TBG is less than TGDH which depending on height of key tree, but more than Braid groups on IKA.2 at one round according to same reason in leave protocol. The number of rounds in some operation in TBG is more than other protocol, since our protocol have to challenge the maximum number of one-hop neighbors on every current member as director in leave and partition event. Also, in the merge protocol, the merging group members have to challenge the maximum signal strength measuring with members in current group. The number of messages in TBG is totally more than other protocol since the lager number of messages in TBG is used for supporting the discovery director in first round. In TBG, the number of messages in merge and partition operation depend on number of merging members and number of remaining members, respectively in the same reason. The number of messages in TBG is more than other but TBG is fastest transmission information from director to other members.

Table 1 Communication Cost

Protocol	Operation	Rounds	Message	Unicast	Multicast
TGDH [7]	Join	2	3	0	3
	Leave	1	1	0	1
	Merge	2	3	0	3
	Partition	O(log *n*)	O(log *n*)	0	O(log *n*)
Braid groups on IKA.2 [5]	Join	2	2	1	1
	Leave	1	1	0	1
	Merge	*m*+3	*n*+2*m*+1	*n*+2*m*-1	2
	Partition	1	1	0	1
TBG	Join	2	2	1	1
	Leave	2	*n*	0	*n*
	Merge	3	*m* +2	1	*m*+1
	Partition	2	*n*-*p*+1	0	*n*-*p*+1

The computation cost in Table 2, the serial number of modular exponentiations for TGDH is O(log *n*). Otherwise the serial number of braid permutations for Braid group on IKA.2 protocol is O(*n*). TBG, our protocol, the serial number of braid permutations is O(log *n*). Our protocol, TBG, and Braid group on IKA.2 reduce the exponential computation in Diffie-Hellman to linear computation by using braid groups.

Therefore, TBG protocol requires less the computation cost than others. TBG protocol reduces the number of rounds in communication cost to constant round. Since the braid groups, key tree and director selecting process in our protocol admit of improvement in communication and computational efficiency.

Table 2 Computation Cost

Protocol	Operation	Exponentiations	Permutation
TGDH	Join	3/2 log *n*	0
	Leave	3/2 log *n*	0
	Merge	3/2 log *n*	0
	Partition	3 log *n*	0
Braid groups on IKA.2	Join	0	*n*+3
	Leave	0	*n*-1
	Merge	0	*n*+2*m*+1
	Partition	0	*n*-*p*
TBG	Join	0	log (*n*+1)
	Leave	0	log (*n*-1)
	Merge	0	log (*n*+*m*)
	Partition	0	log (*n*-*p*)

6. Conclusion

We propose tree-based group using braid groups key exchange for group communication. The modified TGDH using braid groups support dynamic membership group event including join, leave, merge and partition with forward and backward secrecy. Our protocol involves braid groups operation including product and inverse with key tree whose computation is slower than modular exponentiation in TGDH and braid groups on IKA.2. Our protocol is fully contributory scenario for key agreement that not requires the trust party or long-term controller to avoid the problems with the centralized trust and the single point of failure. Our protocol avoid the member serialization by using key tree, a number of existing protocols require group member sequencing that in mobile ad hoc networks is not efficient since the sequence may not correspond to the best geographic node placement and may lead to increased communication cost. Then communication cost in our protocol less than Braid groups on IKA.2 protocol. Finally our protocol can reduce the computation cost in group event while preserving the constant round communication and the security property. Thus our protocol, TBG, is suitable for environment of mobile ad hoc networks.

References

[1] R. Kui, Y. Gang; Efficient Key Agreement in Ad-hoc Network, 2004.

[2] W. Diffie and M. E. Hellman; New Direction in Cryptography, IEEE Transactions on Information Theory, IT-22, 6, pp 664-654, 1976.

[3] I. Anshel, M. Anshel and D. Goldfeld; An Algebraic Method for Public-Key Cryptography, Mathematical Research Letters 6, pp 1-5, 1999.

[4] I. Anshel, M. Anshel and B. Fisher, New Key Agreement Protocols in Braid Group Cryptography, RSA, 2001.

[5] K. H. Ko, S. J. Lee, J. H. Cheon, J. W. Han, J. Kang and C. Park; New Public-Key Cryptosystem Using Braid Groups, Proceedings of Crypto 2000, LNCS 1880, pp166-183, 2000.

[6] H. K. LEE, H. S. LEE and Y.R. LEE; An Authenticated Group Key Agreement Protocol on Braid groups, eprint 2003.

[7] Y. Kim, A. Perrig, and G. Tsudik.; Tree-based Group Key Agreement, ACM Transactions on Information and System Security, 7(1), pp 60–96, 2004.

[8] D. Wallner, E. Harder, and R. Agee; Key Management for Multicast: Issue and architecture, Internet-Draft draft-wallner-key-arch-00.txt, 1997.

[9] Yan Sun and K. J. Ray Liu; Scalable Hierarchical Access Control in Secure Group Communications,

[10] R. J. Hwang, R. C. Chang and K. J. Lin; Key Agreement in Ad hoc Networks, International Conference of Supercomputing, 2002.

[11] K. Mahlburg; An Overview of Braid Group Cryptography Notes, 2004.

[12] M. J. Compagna; Algorithms in Braid Groups, Cryptology ePrint Archive, 2003.

[13] M. Steiner, G. Tsudik, and M. Waidner; Key Agreement in Dynamic Peer Groups, IEEE Transaction on Parallel and Distributed System, Vol. 11, No. 8, pp 769-780, 2000.

[14] L. Liao, and M. Manulis; Tree-Based Group Key Agreement Framework for Mobile Ad-Hoc Networks, Proceedings of the 20th International Conference on Advanced Information Networking and Applications, Vol. 2, pp 5-9, 2006.

[15] Y. Kim, A. Perrig, and G Tsudik; Communication- Efficient Group Key Agreement, Proceedings of 17th International Conference on Information Security IFIP SEC, pp 229-244, 2001.