Efficient Group Key Agreement on Tree-based Braid Groups

Thanongsak Aneksrup & Pipat Hiranvanichakorn
National Institute of Development Administration
School of Applied Statistics, Bangkok, Thailand
E-mail: tnongs@yahoo.com and pipat@as.nida.ac.th

Abstract

The security issues in ad hoc network are increasingly important. In this paper, we propose a distributed key management what approach by using braid groups and key tree. Without any assumption of prefixed trust relationship between nodes, the proposed method works in a self-organizing way to provide the key management services. The using of the proposed tree-based braid groups has following advantages: (1) the communication cost is minimized to constant time; (2) the complexity of computation is decreased to linear permutation by avoiding modular exponential operation. Our approach is more simple, secure and efficiency for group key management in MANET.

Keywords: Group key agreement, Braid groups, Cryptographic, Mobile ad hoc network, Conjugacy search problem

1. Introduction

The mobile ad-hoc network (MANET), what without the any fixed of infrastructure such as access points and wireless routers, is special case of wireless network. The security solutions are more challenge than wired and wireless networks, because network topology is frequently changed according to the movement of mobile devices. The most of MANET communications are multicast; therefore it is necessary to provide support for secure group communication.

Secure group communication requires scalable and efficient group membership management with appropriate access control measures to protect data and to cope with potential compromises. To this end, a secret key for data encryption must be distributed securely and efficiently to current members. The group key must be changed to ensure backward and forward secrecy when membership topology changed. Several proposals for group key management have been made recently. They range from key distribution schemes for large-scale single-sender multicast to contributory key agreement schemes for small any-to-any peer groups. Although most of them focus on wired networks, extensions to wireless networks including MANET should be explored as such networks are becoming more common place. Due to the lack of fixed infrastructure and limited resources, it will be much more complex to adapt protocols and other technologies from the infrastructure based networks.

1.1 Our Contribution

The purpose of our research is to design an efficient key agreement protocol for a group communication in mobile ad hoc networks environment. Our protocol only secure against a powerful passive adversary who can intercept any broadcast message over public channel without authentication. There are two techniques to be implemented in our protocol including braid groups cryptographic and key tree. The braid groups cryptographic is used to decrease computation cost and key tree is used to reduce communication cost. The protocol is designed to be contributory key agreement without trusted third party or permanent controller. The members generate the shared group key in contributory manner. Furthermore, the radio signal strength is applied to reduce communication time. The nearest node between existing group members and a new member is leader of the group at that moment. The message can be fastest transferred to each other. Moreover our protocol, the modified STR using braid group support dynamic membership group operations including join, leave, merge, partition and key refreshing to satisfy security requirement including group key secrecy, forward secrecy, backward secrecy and key independence.

1.2 Related Works

Key establishment using contributory key agreement demands each member to contribute for the group key generating. This scheme is fault tolerant to avoid the problem with the centralized trust and single point of failure. The protocols are discussed as follows. Anton and Duarte (2002) discussed a number of such protocols previously used on wired networks and concluded that the CLIQUES (Steiner, Waidner, and Tsudik, 1998) protocol suite is best suited for ad hoc networks. Li, Wang, and Frieder (2002) also used the GDH (Group Diffie-Hellman) protocol, part of the CLIQUES protocol suite, for key agreement over ad hoc networks. GDH

was an efficient protocol with good support for member join and leave operations but it had some unfavorable features with regard to ad hoc networks. Most importantly, the GDH scheme required that the members be serialized or structured in order to compute the group key. Also, the last member in the group acted as a Group Controller (GC). Consequently the GC was more computation than the other members in the group. Thus, in using GDH for ad hoc networks deciding which member is going to perform the operation of a GC was an important problem.

Kim, Perrig, and Tsudik (2000) adapted GDH to a distributed key agreement protocol TGDH for reducing computation cost of GDH protocol. The protocol built the keys in key tree, which every node on the key tree has a Diffie-Hellman key pair. The number of message in partition event in the order of log(N) is important problem of their approach. Another group key agreement developed for teleconferencing was proposed by Steer, Strawczynski, Diffie and Wiener (1988). This protocol is of particular interest since the structure of its group key form a special case of the TGDH. STR is efficient for joining new group members as it takes only two rounds and two modular exponentiations. Member leaving, however, is relatively difficult. Due to the small number of rounds, which results in a low communication overhead, Kim, Perrig, and Tsudik (2001) extended the STR protocol. The proposed constructed protocol that supported dynamic group. However, its computation costs are quite expensive but communication costs are constant round on all membership events and are not depending on the amount of members. The main disadvantage of TGDH and STR protocol is single point of failure at sponsor what had to existing in key tree. These protocols do not support this situation.

The original protocol of BD, CLIQUES, STR and TGDH were optimized by Manulis (2005) for MANET environment to achieve better communication, computation and memory cost. Augot, Bhaskar, Issarny, Sacchetti (2005) proposed two rounds group key agreement based on decisional Diffie-Hellman assumption to secure against a passive adversary and extended into three rounds to secure against an active adversary. The protocol what secures against active adversary and satisfies perfect forward secrecy under decisional Diffie-Hellman assumption was proposed by Nam, Lee, Kim, and Won (2005). The amount of computation is fixed for mobile participants due to the server with sufficient computation power that assumed to environment is responsibility for most of computations. Therefore their protocol is not suitable for homogenous networks with only low power devices. The spanning tree with Deffie-Hellman was applied in group key agreement protocol by R. Rahman and L. Rahman (2008). The queue-based group key Diffie-Hellman (QGDH) was proposed by Hong (2009). The low performance devices were filtered out by queue structure on power of computing and network latency. The higher performance devices are assigned to participate in the construction of group key and then broadcast it to the lower performance members.

Braid groups what was introduced by I. Anshel, M. Anshel and Fisher (2001) changed the concept on number theory that widely implemented in cryptographic. Several researches proposed public key cryptosystem using braid groups based on the hardness of conjugacy problem. The computation cost of braid groups can decrease to number of permutation on linear algebra rather than number of exponential in traditional protocols. Birman, Ko and Lee (1998) and Ko, Lee, Cheon, Han, Kang and Park (2000) have proposed the key exchange protocol on braid groups that based on conjugacy problem in Diffie-Hellman scheme (GDH), so called Ko-Lee problem. The proposed method is different from widely used cryptosystems on number of theory. Kui and Gang (2004) designed protocol on ad hoc networks with dynamic operation protocol composed of join, leave, merge, partition and refresh protocols. Most importantly, the protocol required that the members be serialized to construct the group key similar as GDH protocol.

1.3 Outline

This remainder section is arranged as follows. We mention the background of braid groups in section 2. We present the tree-based group key agreement protocol on braid group including the dynamic event in section 3. We analyze the protocol in security and complexity in section 4 and 5 respectively. The last section, we end the paper with conclusion.

2. Braid Groups

2.1 Definition of braid groups

Emil Artin presented the braid groups systematically in 1925. The n strand braid groups is denoted as B_n. The integer n is called the *braid index* and 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. In Artin generator 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}. The multiplication of two braid words, ab, is the braid achieved by positioning b on the bottom of a. The identity is braid which is not

intertwining strings. Any *n*-braid can be expressed as a *braid word*, e.g., $\sigma_3\sigma_2\sigma_1^{-1}\sigma_2^{-1}$. The inverse of braid word is constructed by reversing each crossing sequentially. For example, if $a = \sigma_3\sigma_2\sigma_1^{-1}\sigma_2^{-1}$, then $a^{-1} = (\sigma_3\sigma_2\sigma_1^{-1}\sigma_2^{-1})^{-1} = \sigma_2\sigma_1\sigma_2^{-1}\sigma_3^{-1}$.

The artin generator relation of *n*-braid groups B_n are as follows:

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

(2) $\sigma_i \sigma_{i+1} \sigma_i = \sigma_{i+1} \sigma_i \sigma_{i+1}$ where $| i - j | = 1$

2.2 Hard problem in the braid groups

In this section, we explain braid groups in generalized conjugacy search problem (GCSP) (Kim, Perrig, and Tsudik, 2004) that is applied in our protocol in order to increase strength of the 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 $\sigma_1, \sigma_2, \ldots, \sigma_{m-1}$. The hardness in GCSP is follows:

Given the pair $(x, y) \in B_n \times B_n$ such that $y = a\ x\ a^{-1}$ for some $a \in B_n$

Objective is to find $b \in B_m$ such that $y = b\ x\ b^{-1}$ for $m \leq n$.

Therefore we can conclude that x and y are conjugate. It is easy to compute y when known a and x but the attacks need exponential time to compute b from $b\ x\ b^{-1}$ when known x and y.

3. Tree-based Group Key Agreement on Braid Groups

The most of existing group key agreement based on Diffie-Hellman protocol what the computation costs are expensive. The researchers attempted to decrease the number of communication rounds for group members. Our protocol is designed based on braid groups cryptographic (Ko et al., 2000) in order to reduce computation costs to linear algebra and based on tree-based group key agreement (Steer et al., 1998) in order to decrease the number of communication rounds to constant. Our technique based on generalized conjugacy search problem that mentioned above. The protocol is considered to limit computing, storage and power capacities in ad hoc network. We describe these techniques in following sections.

3.1 Key tree notation

Key tree is earliest proposed by Wallner, Harder and Agee (1997) and emerged in group key agreement by Kim et al. (2000). The tree structure is widely implemented to decrease the communication, computation and storage overhead. Key tree is implemented in our protocol to suitable solution for contributory group key agreement in mobile ad hoc network. We describe the notation and definition of key tree in follows. The sample of key tree based on STR is shown in Figure 1. The binary tree, every node is either a leaf or a parent of two nodes, is used in key tree. Each node is represented as $[h, v]$ what is associated with a secret key $K_{[h, v]}$ and a blinded key $BK_{[h, v]}$. The blinded key is calculated as $f(K_{[h, v]})$ where function $f(\ )$ is based on braid groups key exchange that we describe in next section. The member is located at the leaf node. The information of each intermediate node, key and blinded key, is computed from the information of two child nodes to achieve the subgroup key. The leaf node M_i, where $1 \leq i \leq n$, knows every key along the path from node M_i to root node, this path is called the *key-path* . In Figure. 1, M_1 knows every key $\{ K_{[3,0]}, K_{[2, 0]}, K_{[1, 0]}, K_{[0, 0]} \}$ in key-path { [3,0], [2,0], [1,0], [0,0] }. The *co-path* is the set of sibling nodes of each node in the key-path of a member M_i. In Figure. 1, the co-path of M_1 is set of node { [3,1], [2,1], [1,1] }. The group secret key is key at the root node, $K_{[0, 0]}$, what can be computed from all blind keys on the co-path and session random $K_{[h, v]}$ of a computing node.

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 consists 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 braids $s_l \in B_{gl}$ and $s_m \in B_{gm}$ with $l \neq m$, $s_l s_m = s_m s_l$. The properties of braid groups are applied in our key exchange protocol as follows.

The $\beta_{[h,v]} \in B_q$, where $B_q \subseteq B_g$, be a sufficiently complicated braid are selected and published. Each member selects $\beta_{[h,v]}$ and publishes as public braid word at the leaf node. The $\beta_{[h,v]}$ at intermediate nodes (parent node) including root node are equal to $\beta_{[h+1,2v]}\beta_{[h+1,2v+1]}$. Supposing *n* members need to share a key. Each member selects the secret key from own braid groups. The blinded key $BK_{[h,v]}$ is generated by $f(K)$ that is equal $K_{[h, v]}\beta_{[h-1, v]}K^{-1}_{[h, v]}$. Therefore key at intermediate nodes $K_{[h, v]}$ are computed as follows:

$$K_{[h, v]} = K_{[h+1, 2v]}\, BK_{[h+1, 2v+1]}\, K^{-1}_{[h+1, 2v]}$$

$$= K_{[h+1, 2v]}\, K_{[h+1, 2v+1]}\, \beta_{[h, v]}\, K^{-1}_{[h+1, 2v+1]}\, K^{-1}_{[h+1, 2v]} \quad \text{or}$$

$$K_{[h, v]} = K_{[h+1, 2v+1]}\, BK_{[h+1, 2v]}\, K^{-1}{}_{[h+1, 2v+1]}$$

$$= K_{[h+1, 2v+1]}\, K_{[h+1, 2v]}\, \beta_{[h, v]}\, K^{-1}{}_{[h+1, 2v]}\, K^{-1}{}_{[h+1, 2v+1]}$$

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

The conclusion of recursive equation is shown as follows:

Base step : for $[h,v]$ which is leaf node,

$K_{[h, v]} = s_{[h, v]}$ where $s_{[h, v]}$ is session random key of member at leaf node $[h,v]$

$BK_{[h, v]} = s_{[h, v]}\, \beta_{[h\text{-}1, 0]}\, s^{-1}{}_{[h, v]}$

Recursive step : for $[h,v]$ which is an intermediate node

$K_{[h, v]} = K_{[h+1, 2v]}\, BK_{[h+1, 2v+1]}\, K^{-1}{}_{[h+1, 2v]} = K_{[h+1, 2v+1]}\, BK_{[h+1, 2v]}\, K^{-1}{}_{[h+1, 2v+1]}$

$BK_{[h, v]} = K_{[h, v]}\, \beta_{[h\text{-}1, v]}\, K^{-1}{}_{[h, v]}$

The key generating at $[h, v]$ requires the information composed of key of one child and blinded key of another child. The root key is group secret key that is shared by all current members. A group key can be computed from each member's secret key and all blind keys on the co-path to the root.

We show in an 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 as shown in Figure.2. 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 $\in B_c$. The B_a, B_b and B_c are different braid groups then we yield $ab = ba$ and $a^{-1}b^{-1} = b^{-1}a^{-1}$. Furthermore, $abc = cba$ and $a^{-1}b^{-1}\, c^{-1} = c^{-1}b^{-1}a^{-1}$

Each member can generate the group key K_{ABC} in contributory manner by recursive equation to achieve as follows:

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

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

Our key tree scheme based on STR protocol (Steer et al., 1988) that each node can compute each intermediate key from own secret key and blinded keys of the co-path nodes, therefore the member at leaf node can compute all keys on the key-path. This instance shows that the member need not to know all blinded keys for generating the group key but knowing the all blinded keys in our protocol in each member is provided for membership change to be more efficient and robust.

The most of past researches was designed based on position of member in key tree. The scheme may be multi-hop communication between new member and the leader of current members. In other words, some instance new member may contact with the leader that is longest distance comparing with other current members. The communication time between new member and leader was longest. Therefore, in our protocol, we use maximum signal strength for communication in shortest range between new member and leader. The leader in this paper is called as **"director",** therefore the director is assigned momentary dynamic event in order to avoid the single point of failure at existing director. The signal strength achieves from embedded hardware in mobile device such as 802.11b/g. This technique can reduce communication time and transfer information from new member to director as fast as possible.

The following section, we describe the protocol that constructs the group key management. Our protocol includes the following operations:

Join: a new member requests to join the group

Leave: a current node requests to leave the group

Merge: a group requests to merge with the current group

Partition: a subset of members request to split from the current group

Key refreshing: a current member requests periodically key refreshed

The notations in protocol are the new key tree containing all blinded keys, number of current group members, number of merging group members are denoted as $T^*[BK]$, n and m , respectively.

3.3.1 Setup Protocol

The members who want to form a group can be ordered according to some criteria such as MAC address of device. The structure of the key tree can be then derived from this order. The first member in the order is selected as director. The blinded key of member M_i is $BK_i = s_i \beta_r \beta_i s_i^{-1}$where β_r is existing publish braid word at root node before the director will update next member to key tree by order. Each member knows the own β_r because it have some criteria such as MAC address of all members. It can order the MAC address by itself, and then it knows sequence of member. The process is illustrated as follows:

Step 1: Each M_i, $i \in \{1, \cdots, n\}$ send its blinded session random key to director M_d.

$$\{ M_i , i \in [1, n] \} - M_d \xrightarrow{BK_i} M_d$$

Step 2: The director computes recursively keys and blinded keys to the root, broadcast the new key tree containing the all blinded keys.

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

Step 3: Each member computes the secret group key.

Then total communication message in setup protocol is *n* rounds including the setup message from each member to director and key tree information from director to all members.

3.3.2 Join Protocol

The group has *n* members, $\{M_1,..,M_n\}$. Every member in current group knows the existing key tree. The new member M_{n+1} wishes to join the group by detecting the maximum signal strength of current group member to be as director in order to communicate in one hop and shortest distance. Later the new member sends, JOIN_MESSAGE, request message to director. The director refreshes the own session random key, computes keys and blinded keys of intermediate nodes up to the root node, and sends the existing key tree to new member. In our protocol, the insertion point of new member on key tree will be new root node because the new member can be computed the information of new key tree with the lowest computation cost. The new member needs to compute only the blinded key at the new root node. Later, the new member updates existing key tree in accordance with creates a new root node and a new member node. Next, the new member selects session random key (i.e., secret key) and computes keys and blinded keys going up to the root. The blinded key of new member M_{n+1} is $BK_{n+1} = s_{n+1}\beta_r \beta_{n+1}(s_{n+1})^{-1}$where β_r is existing publish braid word at root node that the new member can find in existing key tree information. The new member broadcasts the new key tree containing only blinded keys to all other members. Finally all other members compute the new group key. This join protocol provide key independence since director updated session random key that knowledge of a previous group key cannot be used to compute the new group key. Figure 3 and Figure 4, before and after join operation respectively, show an example of M_4 joining a group where director as M_2. This instance, it means that M_2 is nearest with M_4. The conclusion of join protocol is illustrated as follows:

Step 1: The new member detects the maximum signal strength of current group member as director and sends JOIN_MESSAGE request message to join the group. After the director received the request message, it selects its new session random key, computes keys and blinded keys, and sends the exiting key tree to new member.

$$M_d \xrightarrow{T[BK]} M_{n+1}$$

Step 2: The new member selects its session random key, updates key tree, computes keys and blinded keys, and broadcasts the new key tree containing the only all blinded keys.

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

Step 3: Each member computes the secrete group key.

Then total communication message in join protocol is two rounds including existing key tree information from director to new member and new key tree information from new member to all members. There are *n* serial number of braid permutation in the worst case if director is deepest node.

3.3.3 Leave Protocol

We begin with n current members and the member M_r wants to leave the group. In this event the director is the leaf node above the removing node in existing key tree before leave event. In special case, if the leaving node is child of the root, the director is leaf node below the removing node. Because director only calculates the new blinded key of intermediate nodes above director up to the root node, other intermediate nodes are not necessary to update blinded keys. Upon hearing the leave event from the group, the director updates key tree by deleting the leaf node of M_r, selects a new secret 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 members. Then the remainder members compute the new group key. Figure 5 and Figure 6 show before and after leave operation, respectively. It is example of M_2 leaving a group where director as M_3. The conclusion of leave protocol is illustrated as follows:

Step 1: The director selects the new session random key, updates the key tree, computes keys and blinded keys and broadcasts the new key tree.

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

Step 2: Each member computes the group key.

Then total communication message in leave protocol is one round. In the worst case, the serial number of braid permutation in this protocol is equal to n -1 when the leaving node is the deepest leaf.

3.3.4 Merge Protocol

In this instant, we assume that m merging group needs to merge with c current group. The existing merging group director detects to achieve maximum signal strength what is measured as the closest member between itself and current group members. The current group director is the member that has maximum signal strength with merging group director. After the merging process, the leftest leaf of shorter tree becomes the right child of a new intermediate node. The root of the longer tree is left child of the new intermediate node.

After the current group director received the MERGE_MESSAGE message, it refreshes session random key, computes keys and blinded keys, and sends the current group's key tree containing the all blinded keys to merging group director. Later, the merging group director updates key tree by combining the merging group's key tree and current group's key tree at the new root node, the director chooses session random key, computes keys and blinded keys up to the root node, and broadcasts new key tree containing the all blinded keys to all members in new group. Finally, the group key is calculated independently by each member. Figure 7 and Figure 8 show before and after merge operation, respectively. The member that has maximum signal strength of merging group director, M_5, is M_1, and then M_1 is current group director. Then total communication message in merge protocol is two rounds. In the worst case, the serial number of braid permutation in this protocol is equal to m +1 when the merging director is deepest leaf of merging tree.

The conclusion of merge protocol is shown as follows:

Step 1: The director of current group selects new session random key, computes blinded keys and sends update key tree to the merging group director.

$$M_{d[c]} \xrightarrow{T_c[BK]} M_{d[m]}$$

where T_c is key tree of current group.

Step 2: The merging group director selects its new session random key, combines key tree, computes the all blinded keys, and broadcasts the new key tree containing the only all blinded keys.

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

Step 3: Each member computes the group key.

3.3.5 Partition Protocol

The partition operation can occur when a network faults. The partition protocol actually presents a concurrent multiple members leaving from group. When multiple members p need to leave the group, the director is the node above the undermost removing nodes in existing key tree. Otherwise, if the leaving node is child of root and the undermost removing nodes does not exist, the director is leaf node below the undermost the removing

nodes. Because director only calculates the new blinded key of intermediate nodes above director up to the root node, other intermediate nodes are not necessary to update blinded keys. It means that amount of new blinded keys calculation is least. As for the partition protocol, after the director deletes all leaving members from key tree, it selects new session random key, computes keys and blinded keys going up to the root, and broadcasts the key tree with blinded keys to reminder members. Finally, each member computes the new group key. Figure 9 and Figure 10 show an example of partition operation when all members in G_1 see M_4 and M_5 as leaving members, while all members in G_2 see M_1, M_3, and M_6 as leaving members. The director of G_1 is M_4 and that of G_2 is M_6. The conclusion of partition protocol is shown as follows:

Step 1: The director updates the key tree, selects the new session random key, computes keys and blinded keys and broadcasts the new key tree.

$$M_d \xrightarrow{T^*[BK]} \{ M_i , i \in [1, n\text{-}p] \}$$

Step 2: Each member computes the group key.

Then total communication message in partition protocol is one round. The serial number of braid permutation in partition protocol is equal to $n - p$ where p is amount of partition member.

3.3.6 Key Refreshing

Key refreshing in MANET is necessary, since most nodes can be easily compromised due to their mobility and physical vulnerability. Then the key refreshing should be occurred periodically in order to limit exposure due to the loss of keys. Furthermore, the event number limits the amount of ciphertext available to cryptanalysis for given group key. In our protocol the node that needs to refresh the key acts as the director. In similar way of other protocols, the director chooses the new session random key, computes keys and blinded keys up to the root, and broadcasts updated key tree. All members compute the new group key. The conclusion of key refreshing protocol is shown as follows:

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

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

Step 2: Each member computes the group key.

4. Security Analysis

As described above, we can see that group key agreement on tree-based braid groups satisfies forward and backward secrecy. It also satisfies key independence. The passive adversaries are unable to compute future and previous group key although they know all previous key trees 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 keys 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 updates own session random key and causes the change of keys and blinded keys. Therefore A is unable to compute the subsequent group key, because the key tree information is changed. Thus the 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 updates own session random key and causes the change of keys and blinded keys in key-path. Therefore A is unable to compute previously used group key, since A can only compute new group keys due to change key tree information. Therefore our protocol satisfies the backward secrecy.

5. Complexity

5.1 Communication Cost

The communication cost is shown in Table 1. The number of rounds on TBG is constant in all events same as STR that better than Braid groups on GDH protocols on merge event. The number of rounds on merge operation

in Braid groups on GDH depends on number of merging members, but all operation in TBG and STR does not depend on number of members that dynamic movement. The number of rounds in TBG is equal to STR and Braid groups on GDH in join, leave and partition protocol. In merge protocol, the number of rounds in TBG is less than Braid groups on GDH which depend on number of merging members.

5.2 Computation Cost

The computation cost in Table 2, the serial number of modular exponentiations for STR is O(n). The number of braid permutations for braid groups on GDH protocol is O(n). TBG, our protocol, the serial number of braid permutations is O(n) in leave, merge and partition protocols, except join protocol is constant permutation. Our protocol, TBG, and Braid groups on GDH reduce the exponential computation in Diffie-Hellman to linear computation by using braid groups.

6. Conclusions

We propose tree-based group key agreement on braid groups. The modified STR using braid groups supports dynamic membership group operation including join, leave, merge, partition and key refreshing with satisfies forward and backward secrecy. The single point of failure at temporally controller is got rid from our protocol by momentarily assigning the director on each membership event. Our protocol involves braid groups operation including product and inverse. The key tree whose computation cost is much lower than modular exponentiation in STR and braid groups on GDH is applied in our protocol. Also the protocol avoids 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 increase communication cost. The radio signal strength is applied to our protocol for decreasing the communication cost. Therefore communication cost in our protocol is less than braid groups on GDH protocol. Finally our protocol can reduce the computation cost in group event while preserving the constant round communication and the security property. Therefore TBG is suitable for environment of mobile ad hoc networks.

References

D. Augot, R. Bhaskar, V. Issarny and D. Sacchetti. (2005). An efficient group key agreement protocol for ad hoc networks. *Proceedings of the First International IEEE WoWMoM Workshop on Trust, Security and Privacy for Ubiquitous Computing*, 3, 576 – 580.

D. Steer, L. Strawczynski, W. Diffie, and M. Wiener. (1988). A secure audio teleconference system. *Advances in Cryptology – CRYPTO'88*, 520–528.

D.M.Wallner, E.J.Harder and R.C.Agee. (1997). Key management for multicast: Issues and architecture. [Online] Available: http://tools.ietf.org/html/draft-wallner-key-arch-00 (June, 1997)

E.Anton and O.Duarte. (2002). Group key establishment in wireless ad hoc networks. *Workshop on Quality of Service and Mobility (WQoSM)*.

I. Anshel, M. Anshel and B. Fisher. (2001). New key agreement protocols in braid group cryptography. *Proceedings of the CT-RSA 2001*, 1-15.

J. Birman, K. H. Ko and S. J. Lee. (1998). A new solution to the word and conjugacy problems in the braid groups. *Advances in Mathematics 139*, 322-353.

J. Nam, J. Lee, S. Kim, and D. Won. (2005). DDH-based group key agreement in a mobile environment. *Journal of Systems and Software*, 78(1), 73–83.

K. H. Ko, S. J. Lee, J. H. Cheon, J. W. Han, J. Kang and C. Park. (2000). New public-key cryptosystem using braid groups. *Proceedings of Crypto 2000*, 166-183.

M. Manulis. (2005). Contributory group key agreement protocols, revisited for mobile ad-hoc groups. *IEEE International Conference on Mobile Ad hoc and Sensor Systems Conference,* 818-825.

M. Steiner, M. Waidner, and G. Tsudik. (1998). Cliques: A new approach to group key agreement. *Proceedings of the 18th International Conference on Distributed Computing Systems*, 380-387.

R. H. Rahman and L. Rahman. (2008). The new group key management protocol for wireless ad-hoc networks. *International Journal of Computer and Information Science and Engineering*, 2(2), 74-79.

R. Kui and Y. Gang. (2004). Efficient key Agreements in ad-hoc networks. *Proceedings of 8th Conference of China Cryptography*.

S. Hong. (2009). Queue-based group key agreement protocol. *International Journal of Network Security,* 9(2), 135-142.

X. Li, Y. Wang, and O. Frieder. (2002). Efficient hybrid key agreement protocol for wireless ad hoc networks. *Proceedings of 11th International Conference on Computer Communications and Networks*, 404-409.

X.Y. Li, Y. Wang, and O. Frieder. (2002). Efficient hybrid key agreement protocol for wireless ad-hoc networks. *Proceedings of 11th IEEE International Conference on Computer Communications and Networks (ICCCN02)*, 404-409.

Y. Kim, A. Perrig, and G. Tsudik. (2000). Simple and fault-tolerant key agreement for dynamic collaborative groups. *7th ACM Conference on Computer and Communications Security*, 235–244.

Y. Kim, A. Perrig, and G. Tsudik. (2001). Communication-efficient group key agreement. *Proceedings of the 17th International Information Security Conference* (*IFIP SEC01*), 229-244.

Y. Kim, A. Perrig, and G. Tsudik. (2004). Tree-based group key agreement. *ACM Transactions on Information and System Security*, 7(1), 60-96.

Y. M. Tseng. (2005). A robust multi-party agreement protocol resistant to malicious participants. *The Computer Journal*, 48, 480-487.

Table 1. Communication Cost

Protocol	Operation	Rounds	Message	Unicast Message	Multicast Message
STR	Join	2	2	1	1
	Leave	1	1	0	1
	Merge	2	3	2	1
	Partition	1	1	0	1
Braid groups on GDH	Join	2	2	1	1
	Leave	1	1	0	1
	Merge	m+3	n+2m+1	n+2m-1	2
	Partition	1	1	0	1
TBG	Join	2	2	1	1
	Leave	1	1	0	1
	Merge	2	2	1	1
	Partition	1	1	0	1

Table 2. Computation Cost

Protocol	Operation	Exponentiations	Permutation
STR	Join	2	0
	Leave	$3n/2 + 2$	0
	Merge	$2m$	0
	Partition	$3n/2 + 2$	0
Braid groups on GDH	Join	0	n+3
	Leave	0	n-1
	Merge	0	n+2m+1
	Partition	0	n-p
TBG	Join	0	n
	Leave	0	n-1
	Merge	0	m+1
	Partition	0	n-p

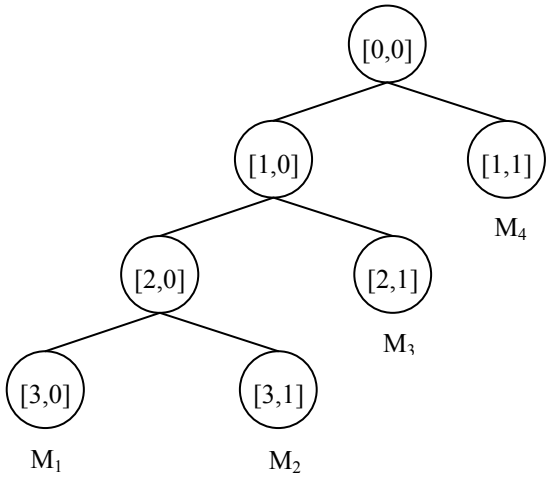

Figure 1. Notation of key tree

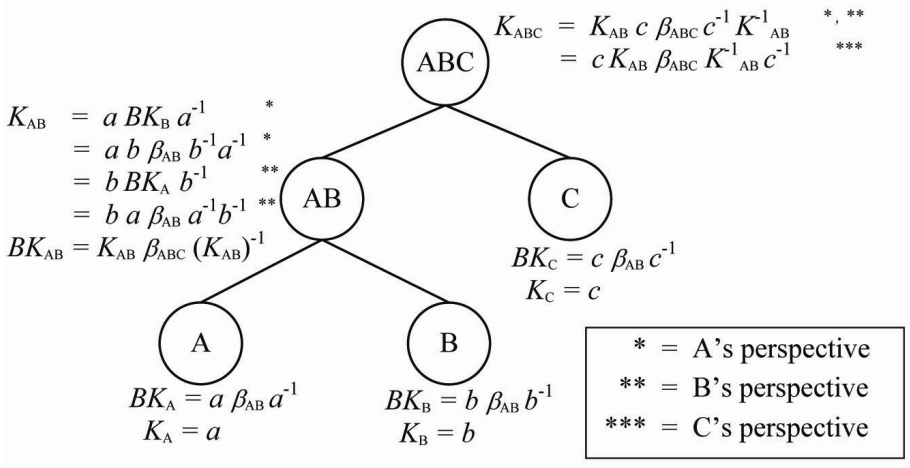

Figure 2. Group Key Generating

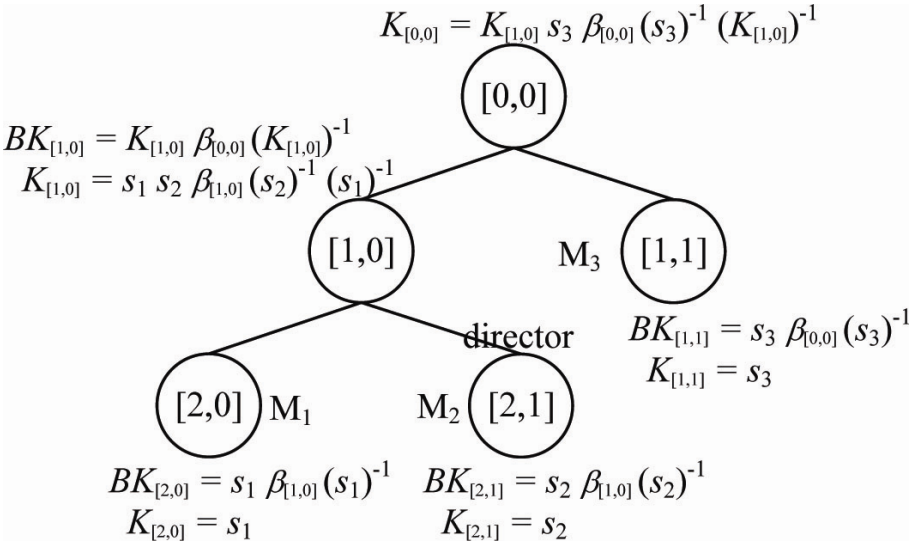

Figure 3. Before tree update in join protocol: M_4 join, M_2 as director

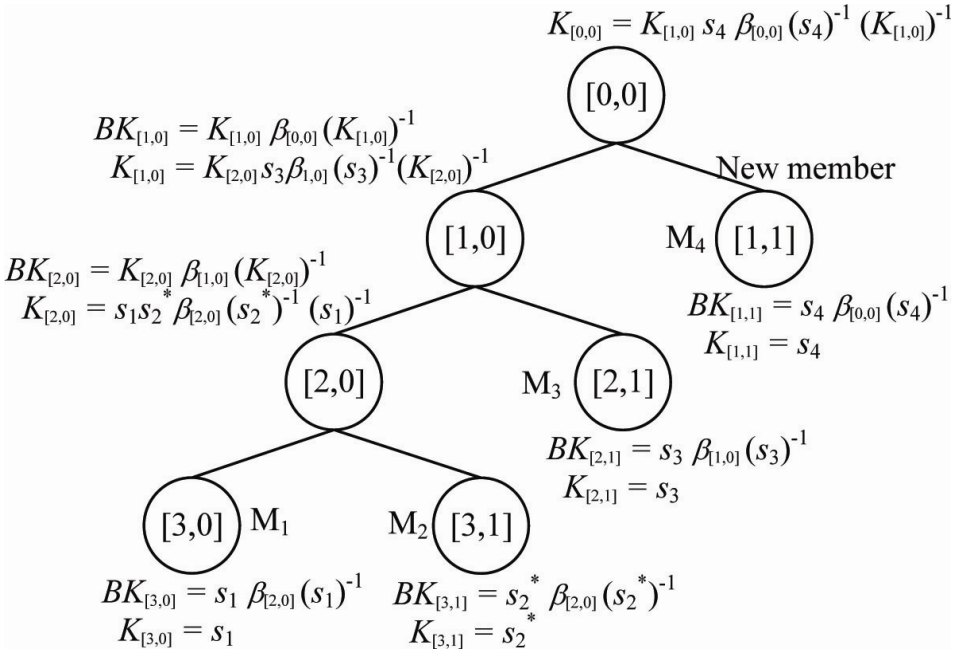

Figure 4. After tree update in join protocol: M_4 join, M_2 as director

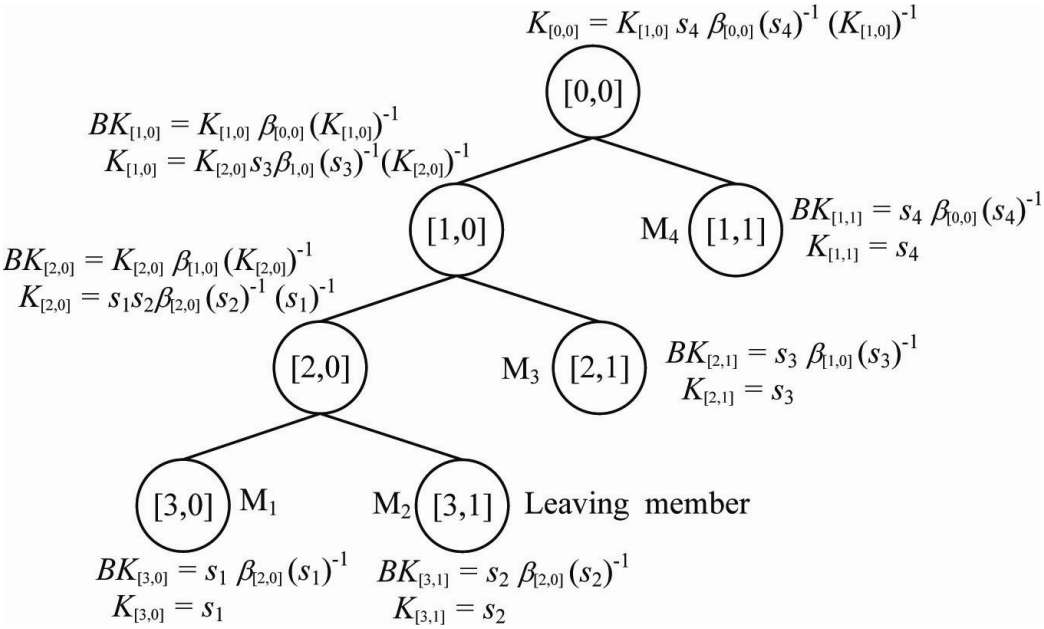

Figure 5. Before tree update in leave protocol: M_2 leave, M_3 as director

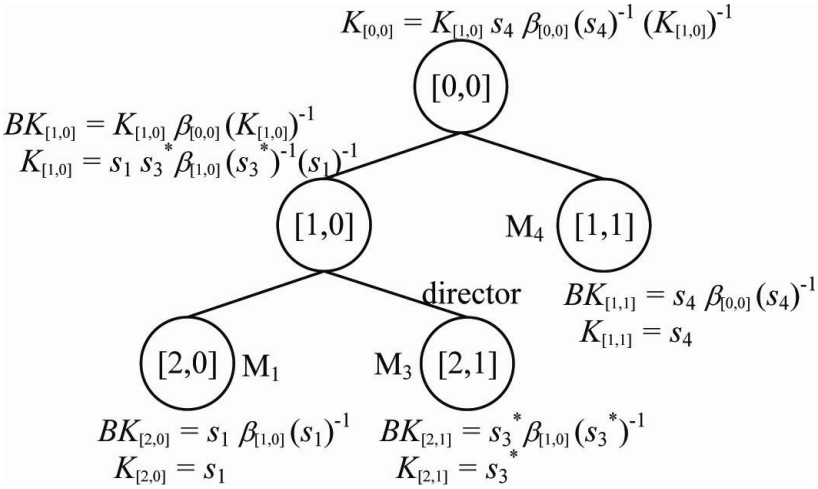

Figure 6. After tree update in leave protocol: M_2 leave, M_3 as director

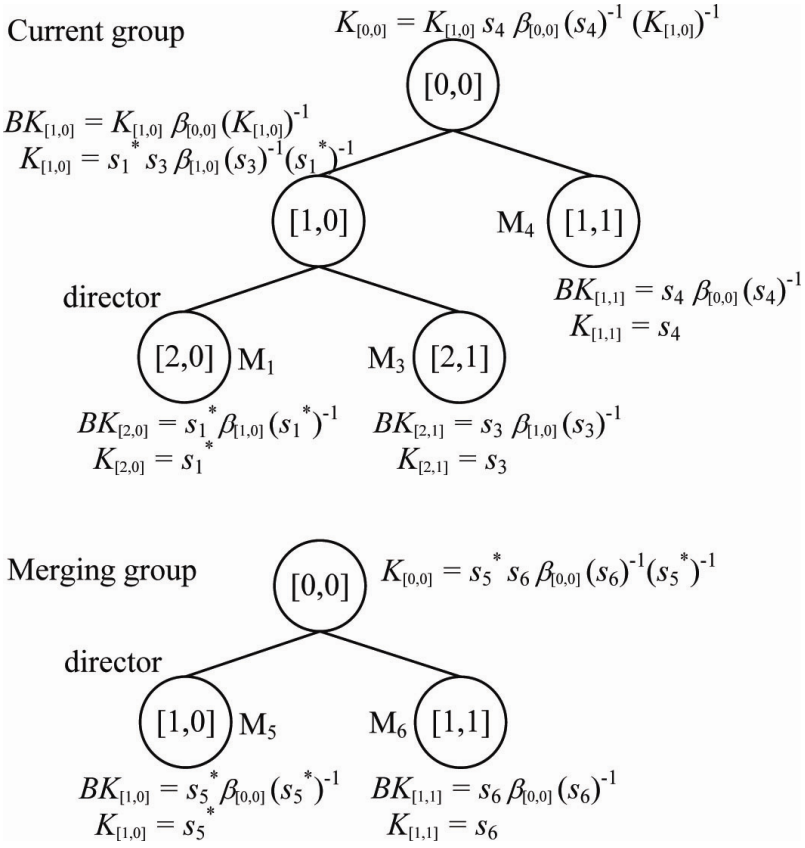

Figure 7. Before tree update: Merge Protocol

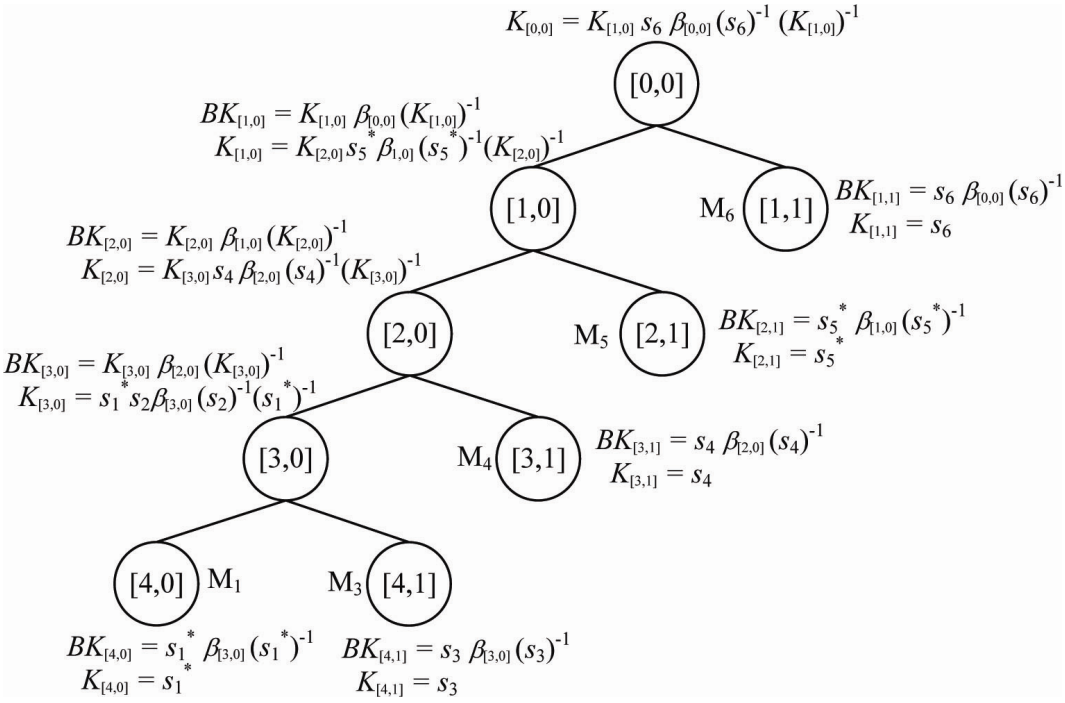

Figure 8. After tree update: Merge Protocol

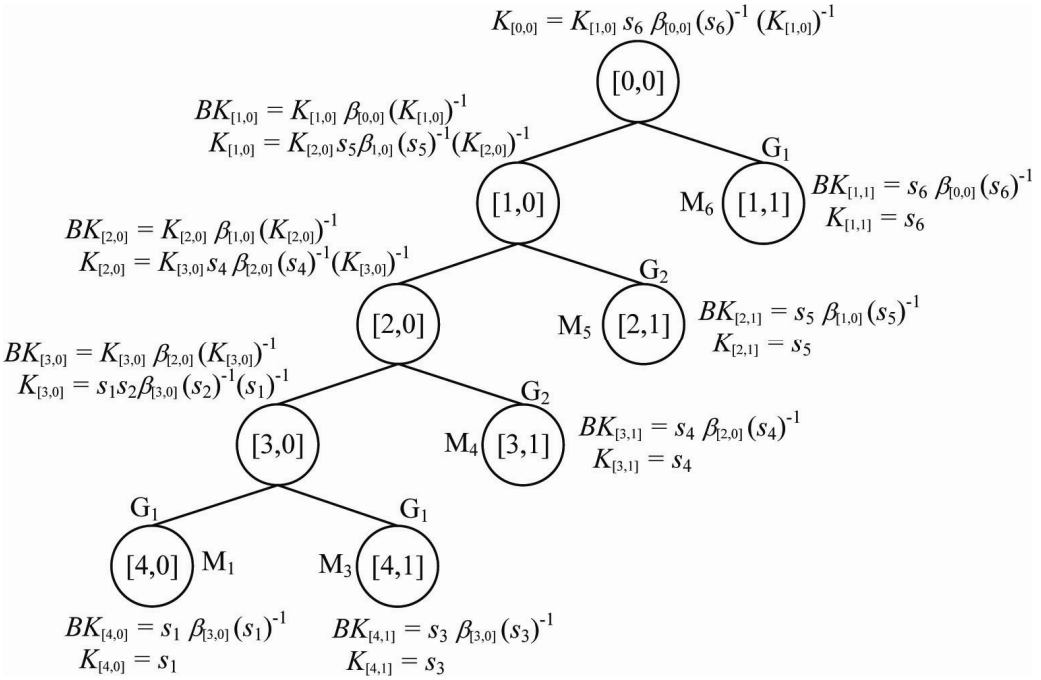

Figure 9. Before tree update: Partition Protocol

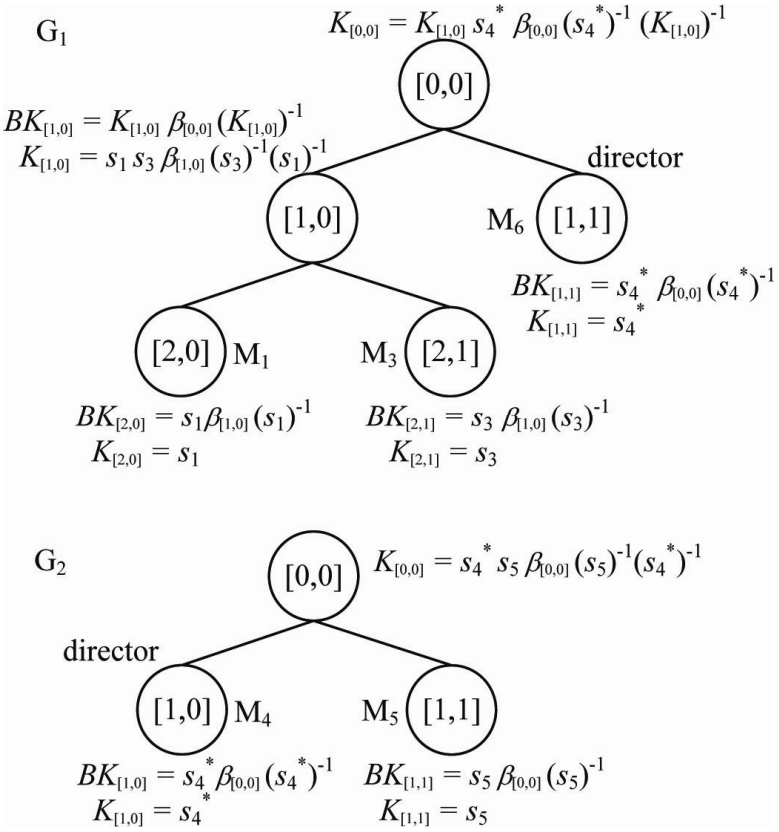

Figure 10. After tree update: Partition Protocol