Broadcast Encryption Based on Braid Groups

Norranut Saguansakdiyotin and Pipat Hiranvanichakorn

National Institute of Development Administration
School of Applied Statistics, Bangkok, Thailand

Summary
Broadcast encryption is the scheme that a sender encrypts messages for a designated group of receivers, and sends the ciphertexts by broadcast over the networks. Many research papers have done it using elliptic curve cryptography. In this paper, we propose the broadcast encryption scheme based on braid groups cryptography which is an alternative method in the public key cryptography and can reduce the computational cost. Our scheme has some advantages over the scheme using symmetric group key in that the sender can be someone inside or outside the group and it gets rid of the problem in distributing a secret key.

Key words:
Broadcast encryption, Braid groups, Asymmetric group key agreement

1. Introduction

A. Fiat and M. Naor [1] first proposed the concept of broadcast encryption in 1993. In this scheme, sender allows to send a ciphertext to some designated groups whose members of the group can decrypt it with his or her private key. However, nobody outside the group can decrypt the message.

Broadcast encryption is widely used in the present day in many aspects, such as VoIP, TV subscription services over the Internet, communication among group members or from someone outside the group to the group members. This type of scheme also can be extended in networks like mobile multi-hop networks, which each node in these networks has limitation in computing and storage resources.

There are many research papers about broadcast group-oriented encryption as in C. Ma, Y. Wu, and J. Li [2] and C. Ma and J. Ao [3]. The former proposes a novel broadcast encryption used in the group communication. It is an asymmetric group key agreement scheme achieved a broadcast message with constant ciphertexts and private keys. The later proposes the improved version by including the identity of users to the previous scheme, and it is secure against chosen ciphertext attack and the key generation withstands collude attack from the users of the group. Both of the schemes as mentioned before are implemented using elliptic curve cryptography (ECC).

Braid groups were introduced in 1947 by E. Artin [4] and first used to construct a Diffie-Hellman type key agreement protocol and a public key encryption scheme by K. Ko et al [5]. In P. Karu and J. Loikkanen work [6], the comparison of fast public key cryptosystems; ECC, NTRU, and braid groups has been made. The result shown that the braid groups based efficient cryptosystem can be implemented, and it is faster than RSA and ECC. For very limited environments like PDA's, smart cards, and mobile phones they require faster cryptosystem, therefore braid groups based cryptography can be one of the choices. The braid groups can be used in a group key agreement protocol as in T. Aneksrup and P. Hiranvanichakorn [7]. In that paper, they propose a contributory symmetric key management protocol using braid groups and key tree. The concept of braid groups assists to avoid modular exponential operation in computation cost and the key tree helps in reducing the communication cost to constant round, so the computation cost and the communication cost can be minimized. There are some research papers in doing asymmetric group key agreement as in Q. Wu, Y. Mu, W. Susilo, B. Qin, and J. Domingo-Ferre [8] and X. Zhao, F. Zhang and H. Tian [9]. The former scheme is constructed on one round asymmetric group key agreement (ASGKA) based on the concept of aggregatable signature based broadcast (ASBB) by using bilinear pairings. An ASBB is the scheme that the public key can be used to verify signatures as well as to encrypt messages, and any valid signature can be used to decrypt the ciphertexts. The later scheme is a dynamic asymmetric group key agreement (DASGKA) combining a conventional authenticated group key agreement, a public key encryption and a multi-signature. Our motivation for this research is to use asymmetric group key in broadcast encryption based on braid groups. Our scheme also achieves a broadcast message with constant ciphertexts and public key. Compares to the schemes in [8] and [9], our scheme uses less communication messages than the schemes in [8] and [9], and it avoids exponential operation as demonstrated in Section 5.

This paper is organized in the following manner. Section 2 gives a brief introduction to braid group based cryptography, hard problems in braid groups, and public

Manuscript received February 5, 2012
Manuscript revised February 20, 2012

key cryptography based on braid groups and then mentions about a notation of a key tree used in our protocols. Section 3 we present our protocol of broadcast encryption based on braid groups. Section 4 states the key management protocols; join and leave protocols related to the key secrecy concept and analyzes the protocol in complexity. We also mention that our protocol can resist collude attack. Section 5 we compare our protocol and other asymmetric group key agreement protocols. The last section, we give a conclusion on this paper.

2. Braid group based cryptography

This section gives a brief introduction to braid groups, some hard problems based on braid groups, and a public key cryptography based on braid groups then the last subsection is a concept of key tree applied in our protocols.

2.1 Braid groups

The braid group was first introduced by E. Artin in 1947. It is a "non-commutative" group which can be used in cryptography because it computations can be performed efficiently, but it is strong enough against attacks. For the geometric presentation of the braid group, a braid B_n is a set of disjoint n strands all of which are attached to two horizontal bars at the top and the bottom, and between the top and the bottom bars, one strand crosses any one horizontal line only once. We call n is the braid index. A braid can be represented by a sequence of generator σ_n which is called the Artin generator as proposed by Artin. If the strand i^{th} passes under the strand $i+1^{th}$, it denotes σ_i. Corresponding if the strand i^{th} passes over the strand $i+1^{th}$ it denotes σ_i^{-1}. The multiplication of two braids with the same braid index, xy comes from concatenating the ends of the strands of the first braid with the beginnings of the strands of the second braid, e.g., $x = \sigma_1^{-1}$and $y = \sigma_2\sigma_1$, so $xy = \sigma_1^{-1}\sigma_2\sigma_1$. The identity braid is the braid consisting of strands with no crossings. The inverse of a braid is the mirror image of that braid with respect to the horizontal line, e.g. From the previous example, $y^{-1} = (\sigma_2\sigma_1)^{-1} = \sigma_1^{-1}\sigma_2^{-1}$.

As we have seen that any braid B_n can be express as a braid word which is a sequence of generator, e.g., $\sigma_1^{-1}\sigma_2\sigma_1$, and it has the following relation;

(1) $\sigma_i\sigma_j = \sigma_j\sigma_i$ where $|i-j| > 1$
e.g., $\sigma_1\sigma_3 = \sigma_3\sigma_1$

(2) $\sigma_i\sigma_j\sigma_i = \sigma_j\sigma_i\sigma_j$ where $|i-j| = 1$
e.g., $\sigma_1\sigma_2\sigma_1 = \sigma_2\sigma_1\sigma_2$

2.2 Hard problems in braid groups

Recently there are some mathematically hard problems in braid groups as a candidate for cryptographic one way function, but the famous one is the generalized conjugacy search problem (GCSP) which we apply it in our protocol to maximize strength of the key. In the generalized conjugacy search problem, it states that x and y are conjugate if there exist an element a such that $y = a\,x\,a^{-1}$ for $m < n$, where B_m is a subgroup of B_n generated by $\sigma_1,\sigma_2,\ldots,\sigma_{m-1}$. The hardness in GCSP is as following;

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

The objective is to find $b \in B_m$ such that $y = b\,x\,b^{-1}$ for $m \le n$.

Thus we can conclude that x and y are conjugate. It is able to compute y easily when we are known both a and x, but needs exponential time to compute b from $b\,x\,b^{-1}$ when known x and y.

2.3 Public key cryptography based on braid groups

This section gives an example in using the braid groups in public key cryptography. We show a simple example protocol, according to P. Dehornoy [10].

Here we say that Alice needs to send message m_A to Bob. Alice uses her private key and Bob's public key to encrypt the message and Bob can use his private key and Alice's public key to decrypt it.

- Alice computes the conjugate $p' = sps^{-1}$, s is Alice's private key and (p,p') is her public key;
- Bob computes the conjugate $p'' = rpr^{-1}$, r is Bob's private key and (p, p'') is his public key;
- Alice sends a ciphertext $m'' = m_A \oplus H(sp''s^{-1})$ together with her public key p' to Bob;
- Bob computes $m_A = m'' \oplus H(r\,p'\,r^{-1})$
  $= m'' \oplus H(r\,sps^{-1}\,r^{-1})$
  $= m'' \oplus H(srpr^{-1}s^{-1})$
  $= m'' \oplus H(s\,p''\,s^{-1})$
  $= m_A$

As shown above, braid r and s commutes with each other, thus $sr = rs$. This property of the braid groups is true when we carefully select the braids. Suppose we have n subgroups $B_{g1}, B_{g2}, B_{g3}, \ldots, B_{gn}$ of g-braid groups where $g = g_1 + g_2 + g_3 + \ldots + g_n$. For any braid $s_l \in B_{gl}$ and $s_m \in B_{gm}$ with $l \neq m$, It is true that $s_l\,s_m = s_m\,s_l$.

2.4 Key tree

A key tree was earliest proposed by D. Wallner, E. Harder, and R. Agee [11] as a tool in centralized group key distribution systems and was adapted by Y. Kim et al. [12] for using in fully distributed, contributory key agreement. T. Aneksrup and P.Hiranvanichakorn [7] also used the key tree of braid groups in symmetric group key agreement. Figure 1 shows an example of key tree as mentioned in [7]. It is a binary tree which has only left subtree. The tree composes of both intermediated and leaf nodes. The root node is located at level *0* and the lowest leaf is at level *h*. Each node is represented as $<l,v>$ where l and v are denoted as v^{th} node at level l in a tree. As shown in Figure 1, a member node M_i where $i \in (1...N)$ is located only at a leaf of the tree. Each node is associated with a key $K_{<l,v>}$ and a blind key $BK_{<l,v>} = f(K_{<l,v>})$ where function $f(K_{<l,v>}) = K_{<l,v>}\ \beta_{<l-1,\ v-1>}\ K^{-1}_{<l,v>}$ where $\beta_{<l-1,\ v-1>}$ is public braid word. For an intermediated node, $K_{<l,v>} = K_{<l+1,2v>}\ BK_{<l+1,\ 2v+1>}\ K^{-1}_{<l+1,2v>}$ or $K_{<l,v>} = K_{<l+1,2v+1>}\ BK_{<l+1,2v>}\ K^{-1}_{<l+1,2v+1>}$ as stated in [7]. A key $K_{<l,v>}$ and a blind key $BK_{<l,v>}$ of an intermediated node is computed independently from the values of key and blind key of child nodes to achieve a subgroup key. In fact, a member node at $<l,v>$ can compute every key along a path from $<l,v>$ to $<0,0>$, referred to as a *key-path*. For example member node M_2 can compute the *key-path*; $<3,1>,<2,0>,<1,0>,<0,0>$.

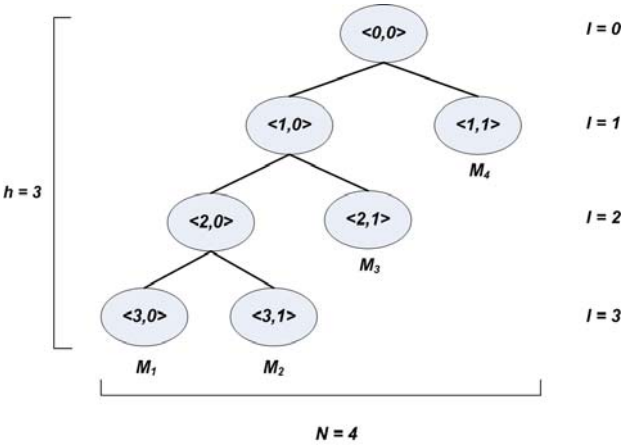

Fig. 1 Notation for key tree.

3. Broadcast encryption based on braid groups

In this section, we review the asymmetric group key agreement (ASGKA) which was introduced by Wu et al., and the dynamic asymmetric group key agreement (DASGKA) which was introduced by Zhao et al. as mentioned in the first section, and then propose our broadcast encryption scheme based on braid groups. In Wu et al. scheme, they propose an asymmetric group key agreement protocol based on aggregatable signature based broadcast (ASBB). An ASGKA protocol has the advantage over a symmetric group key agreement (GKA) protocol in that the ASGKA protocol can verify the sender of a message. Typically in an ASGKA protocol, it has two keys; one is a public group key, which is used as an encryption key for a message to a group and another is a private key, which a group member can use it individually as a decryption key, but in Wu et al. scheme which is based on ASBB, the encryption process is done by using a public group key and the decryption process is done by using a signature of a sender. This signature can be verified by using the public key of that sender. Their scheme does not require any controllers. As mentioned in Wu et al., their scheme does not improve in communication overhead for one-time group applications in which the members of the group are about fully dynamic as in ad hoc networks, because their scheme has heavy communication overhead in key establishment. The Zhao et al. scheme is constructed to fulfill the former scheme by introducing a dynamic asymmetric group key agreement. This scheme supports the environment in which users can join or leave the group efficiently without triggering a new key agreement protocol. There are two significant differences between the scheme in [8] and [9]. The first is that they obtain different decryption key. The decryption key for each member in the former scheme is different but in the later scheme is the same. The second is that the former scheme does not achieve dynamic joining and leaving while the later does. Our scheme is also an ASGKA protocol based on the braid groups based cryptography. We design some protocols which support for the dynamic group broadcast such as join and leave protocols and get better efficiency than the schemes in [8] and [9] as mentioned in Section 5.

Our scheme is made up of three algorithms; setup, encryption, and decryption. In the setup phase, when any user needs to join a group, he sends a join request message to a director. The director is one of the group members and everyone knows a public braid denoted as g. Each user can compute their own public keys PK_i from their private key K_i and the public braid g. We use the key tree mentioned above to construct a public group key. The public group key PK_{Group} can be computed individually from a user private key K_i and other public key according to a position of node in the tree. We are going to state the detail of algorithms in the next section. In the encryption phase, we show that anyone outside a group can send encrypted message to the group members. We demonstrate the decryption method in the decryption phase.

3.1 Setup

We assume that users A, B, and C join a group simultaneously. These users can be ordered according to some criteria such as MAC address or IP address. The first member of a group is the director. Each user has his private key K_i, which is a braid in the different braid groups of each other. In order to setup a group each user needs to compute their own public keys PK_i from their private and a published braid g getting from a broadcast center as the following;

User A: $K_A = a$ where $a \in B_a$ (A's private key)
$PK_A = a\, g\, a^{-1}$ (A's public value)

User B: $K_B = b$ where $b \in B_b$ (B's private key)
$PK_B = b\, g\, b^{-1}$ (B's public value)

User C: $K_C = c$ where $c \in B_c$ (C's private key)
$PK_C = c\, g\, c^{-1}$ (C's public value)

We assume that the order of a group member is users A, B, and C respectively, so user A is the director of the group. At this time a key tree is formed as shown in Figure 2. The director can compute key tree consisting of member public keys PK_i and public group keys PK_{Group}. In this case, the key tree consisting of the values of PK_A, PK_B, PK_{AB}, PK_C, *and* PK_{ABC}, then user A must broadcast this key tree to all member.

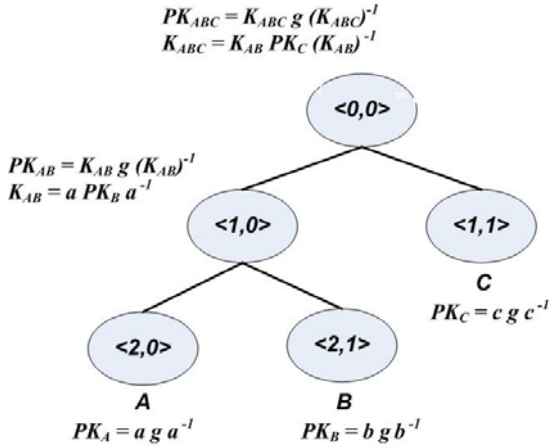

Fig. 2 Group key at setup phase computed by user A.

When every member in the group receives the key tree, they can use this key tree information in the future in order to compute a new public group key PK_{Group}. From the previous scenario, user B and C can also compute the public group keys PK_{Group} if they are selected to be the director as the following;

For user B's point of view;

$$PK_{ABC} = K_{ABC}\, g\, (K_{ABC})^{-1}$$
$$K_{ABC} = K_{AB}\, PK_C\, (K_{AB})^{-1}$$
$$K_{AB} = b\, PK_A\, b^{-1}$$

For user C's point of view;

$$PK_{ABC} = K_{ABC}\, g\, (K_{ABC})^{-1}$$
$$K_{ABC} = c\, PK_{AB}\, c^{-1}$$

In our protocol, each user needs to send his or her public key to the director then the director computes the key tree and broadcast it to all members, thus the total communication messages in key agreement and public group key generation are n+1 multicast messages. The total computation cost is $n + (n\text{-}1)$ serial numbers of braid group multiplication. The n serial numbers of braid group multiplication are for the public key computation of the n members, and the n-1 serial numbers of braid group multiplication are for the public group key computation by director.

3.2 Encryption

In this phase, a user inside or outside the group can send a ciphertext to the group members by encrypting it with the sender private key and the group public key. The receivers, which are the group members, can decrypt it using their own private keys and the sender's public key. We show examples in two cases; the first is that the sender is a group member and the second is that the sender is not a group member.

The first case; a scenario occurs when user C, which is the group member referred from the previous subsection, sends a ciphertext to the group members;

In order to encrypt the message m_C, user C computes the ciphertext m' by encrypting it with user C's private key and the group public key, then he sends m' together with his public key PK_C to the group members as the following;

$PK_C = c\, g\, c^{-1}$

$m' = m_C \oplus H\,(R\, cPK_{ABC}\, c^{-1})$ and sends m', a random braid R and PK_C to members of the group. The random braid R can be changed in every message that sent.

The second case; the following example occurs when user D, which is not the group member referred from the previous subsection, sends a ciphertext to the group members. In order to encrypt the message m_D, user D computes the ciphertext m' by encrypting it with user D's private key and the group public key, then he sends m'

together with his public key PK_D to a group members as the following;

$PK_D = d\ g\ d^{-1}$

$m' = m_D \oplus H\ (R_1\ d\ PK_{ABC}\ d^{-1})$ and sends m', R_1 and PK_D to members of the group.

3.3 Decryption

Each member of the group can decrypt the ciphertext with his own private key. For short, we give an example of user A that is the group member wants to decrypt the message as the following;

For the first case;

$m_C = m' \oplus H\ (R\ K_{ABC}\ PK_C\ (K_{ABC})^{-1})$

$m_C = m' \oplus H\ (R((a\ PK_B\ a^{-1})PK_C\ (a\ PK_B\ a^{-1})^{-1})\ PK_C\ ((a\ PK_B\ a^{-1})\ PK_C\ (a\ PK_B\ a^{-1})^{-1})^{-1})$

For the second case;

$m_D = m' \oplus H\ (R_1\ K_{ABC}\ PK_D\ (K_{ABC})^{-1})$

$m_D = m' \oplus H\ (R_1((a\ PK_B\ a^{-1})PK_C\ (a\ PK_B\ a^{-1})^{-1})\ PK_D\ ((a\ PK_B\ a^{-1})\ PK_C\ (aPK_B\ a^{-1})^{-1})^{-1})$

From the above example, user A can use his private key a in the term K_{ABC} and $(K_{ABC})^{-1}$ as in the user A's view as we mentioned in the previous subsection.

3.4 Correctness

This section shows short correctness of our algorithms for the second case of the examples and omits the first case as the following;

$m_D = m' \oplus H\ (R_1\ K_{ABC}\ PK_D\ (K_{ABC})^{-1})$

$= m' \oplus H\ (R_1\ K_{ABC}\ d\ g\ d^{-1}\ (K_{ABC})^{-1})$

$= m' \oplus H\ (R_1\ d\ K_{ABC}\ g\ (K_{ABC})^{-1}\ d^{-1})$

$= m' \oplus H\ (R_1\ d\ PK_{ABC}\ d^{-1})$

As we have seen above, any user which is not the group member cannot decrypt the ciphertext because he does not know the value of K_{ABC}.

4. Key secrecy

The key secrecy is the concept related to the membership changes. Typically there are two types of the key secrecy; backward and forward secrecy. The backward secrecy prevents a new member joining the group to know the previous key of the group. A new group key distributing to the group members when a new member joins the group cannot be used to decrypt the previous ciphertext. The forward secrecy is used to prevent a leaving member to use the previous key to decrypt a ciphertext. Our scheme fulfils the concept of both backward secrecy and forward secrecy. We show this by using two protocols; join and leave protocols. The join protocol is operated when a new member needs to join a group, on the other hand the leave protocol is operated when a member needs to leave the group.

4.1 Join protocol

In our scheme, when a new member needs to join a group, he will send a request to join a group message to a director. The director can be anyone in the existing group members. After the director receives the join request message, he can generate a new key tree including the new member's public key and new public group key in the tree and then broadcasts this new key tree to all members. The insertion point of a new member in a key tree is at a new root node. From section 3.1, the setup phase, we continue with the scenario when user D needs to join the group as shown in Figure 3. User C as a director can compute a new group key $K_{ABCD} = K_{ABC}\ PK_D\ K^{-1}_{ABC}$ and a new public group key PK_{ABCD}. The new member can also compute the new public group key but he cannot compute the previous group key K_{ABC} because he does not know each user's private key. Thus our scheme complies with the concept of backward secrecy.

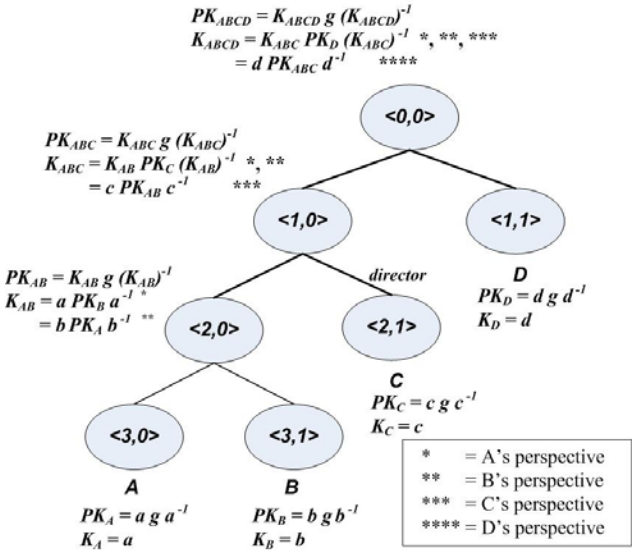

Fig. 3 Key tree after user D joins the group.

The total communication message for join protocol are two multicast messages (include the join request message), and the total computation cost is one serial number of braid group multiplication.

The multiple join occurs when any m users want to join a group simultaneously. In this case the total communication messages are m+1 multicast messages (include the join request messages), and the total computation cost is m serial numbers of braid group multiplication.

4.2 Leave protocol

The leave protocol operates when any users need to leave the group. A leaving member broadcasts a leave request message to all members. For our protocol we design the director is a member above the leaving node in a tree in order to minimize the computation. In a case that the leaving node is a child of root node in the existing tree, the director is a leaf node below the leaving node. The director has to compute a new key tree, and then broadcasts it to all members. We continue with the scenario from the join protocol in section 4.1. In this leave protocol scenario, we assume that user C is going to leave the group. User D will be a director of the group and responsibility to compute a new key tree as shown in Figure 4. In this case, user D computes a new group key $K_{ABD} = d\ PK_{AB}\ d^{-1}$, and a new public group key PK_{ABD}, then broadcasts new key tree to all members. Our protocol designed to comply with the concept of forward secrecy as we state above. For example, the leaving user C cannot know the value of the new group key K_{ABD} and he cannot use his private key to decrypt messages.

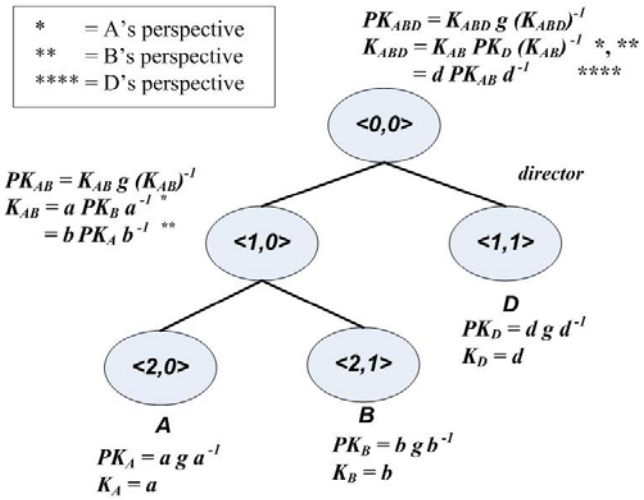

Fig. 4 Key tree after user C leaves the group.

The total communication message for leave protocol are two multicast messages (include the leave request message). In worse case, the number of computation cost in this protocol is equal n-2 when the leaving user is the first one that joins the group.

The multiple leave occurs when any m users want to leave a group simultaneously. In this case the total communication messages are m+1 multicast messages (include the leave request messages), and the total computation cost is n-m-1 serial numbers of braid group multiplication.

4.3 Collude attack

A collude attack can be occurred when two or more users work together and they can forge a valid private key which it will be given to anyone. Our protocol can resist a collude attack like this. The first reason is that in our protocol a private key of user comes from user itself, so it is not distributed from PKG. Users can produce their own private keys and then publish public keys to others. The second is that public and private keys of user are related together e.g. $PK_A = K_A g(K_A)^{-1}$. If someone forges a private key of anyone, so their private and public keys are not related then they can know it.

5. Complexity

In this section we compare our broadcast group-oriented encryption scheme with the scheme proposed by Ma et al., Wu et al. and Zhao et al. in communication and computation cost as in the following subsections.

5.1 Communication cost

The communication cost is shown in Table 1. We analyze the communication cost by comparing both unicast and multicast messages for every member in the system. For join and leave operations, we assume that there are n existing members in a group and m members need to join or leave the group. For Ma et al., they do not state how to publish the public group key and send private key to each member, so we carefully think that it supposes to be unicast message and wrote it down with remark. Another notation in this table that we want to state clearly is that the join and leave operations were not proposed in Ma et al. and Wu et al. protocols. For Zhao et al. we also omit the process for generating a multi-signature because we need to compare it in the same condition with the others.

Table 1: Communication cost

Protocol	*Operation*	*Message*	*Unicast Message*	*Multicast Message*
Ma, Wu, Li [2]	KeyAgree and PKgen	*n* *	*n* *	-
	Join	-	-	-
	Leave	-	-	-
Wu, Mu, Susilo, Qin, Domingo-Ferrer [8]	KeyAgree and PKgen	*n*	-	*n*
	Join	-	-	-
	Leave	-	-	-
Zhao, Zhang, Tian [9]	KeyAgree and PKgen	*2n*	-	*2n*
	Join	*2m+4*	-	*2(m+2)*
	Leave	*2m*	-	*2m*
Our Protocol	KeyAgree and PKgen	*n+1*	-	*n+1*
	Join	*m+1*	-	*m+1*
	Leave	*m+1*	-	*m+1*

*: does not mention clearly

5.2 Computation cost

The computation cost is shown in Table 2. The values in the table are measured in Big-O notation. Our protocol has only multiplication in braid groups while the others have both multiplication in G or G_τ , and also exponentiation.

Table 2: Computation cost

Protocol	*Operation*	*Computation*
Ma, Wu, Li [2]	KeyAgree and PKgen	$O(n)E$
	Join	-
	Leave	-
Wu, Mu, Susilo, Qin, Domingo-Ferrer [8]	KeyAgree and PKgen	$O(n)M + O(n)M_\tau + O(n^2)E + O(n)E_\tau$
	Join	-
	Leave	-
Zhao, Zhang, Tian [9]	KeyAgree and PKgen	$O(n)E$
	Join	$O(n+m)E$
	Leave	$O(n+m)E$
Our Protocol	KeyAgree and PKgen	$O(n)Mul$
	Join	$O(m)Mul$
	Leave	$O(n-m)Mul$

n: the total number of members in the protocol; m: the number of members who want to join/leave the group; G: element in G; G_τ: element in G_τ ; M: multiplication (or division) in G; E: exponentiation in G; M_τ: multiplication (or division) in G_τ; Mul: multiplication in braid groups

6. Conclusion

We propose a broadcast encryption scheme based on braid groups cryptography. Our scheme is asymmetric group key agreement protocol and it is an encryption scheme in which the sender can broadcast an encrypted message over the networks by using his or her private key together with the public group key. The receivers which are the group members can decrypt it with their own private keys together with the public key of the sender. Our scheme makes the constant of ciphertext and public key. The computation cost of our scheme is only one serial number of braid group multiplication when a new member joins the group, and equal to *n*-2 when any member leaves the group.

References

[1] A. Fiat and M. Naor.: Broadcast Encryption. Advances in Cryptology-CRYPTO, Springer-Verlag, Lecture Notes in Computer Science 773: 480-491 (1993)

[2] C. Ma, Y. Wu, and J. Li.: Broadcast Group-oriented Encryption for Group Communication. Communications, Circuits and Systems Proceedings,1623-1626 (2006)

[3] Chunbo Ma., Jun Ao.: Improved Group-oriented Encryption for Group Communication. International Joint Conference on Computational Sciences and Optimization.667-671 (2009)

[4] E. Artin,: Theory of braids, Annals of Math. 48, 101-126, (1947)

[5] K. Ko, S. Lee, J. Cheon, J. Han, J. Kang, and C. Park,: New Public-Key Cryptosystem Using Braid Groups. Crypto' 2000, Lecture Notes in Computer Science 1880, 166-183 (2000)

[6] P. Karu, and J. Loikkanen.: Practical Comparison of Fast Public-key Cryptosystems. Seminar on Network Security, Telecommunications Software and Multimedia Laboratory, Helsinki University of Technology. (http://www.tml.hut.fi/Opinnot/Tik-110.501/2000/papers.html.)

[7] T. Aneksrup and P.Hiranvanichakorn.: Efficient Group Key Agreement on Tree-based Braid Groups. Computer and Information Science, Vol.4, No.1, January (2011)

[8] Qianhong Wu, Yi Mu, Willy Susilo, Bo Qin, and Josep Domingo-Ferre.: Asymmetric Group Key Agreement. EUROCRYPT 2009, Lecture Notes in Computer Science 5479, 153-170 (2009)

[9] Xingwen Zhao, Fangguo Zhang, and Haibo Tian.: Dynamic asymmetric group key agreement for ad hoc networks. Ad Hoc Networks, Volume 9, Issue 5, 928-939 (2011)

[10] Patrick Dehornoy,: Braid-based Cryptography. Contemporary Mathematics, 360 (2004) 5-33

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

[12] Y.Kim, A.Perrig and G.Tsudik.: Simple and Fault-Tolerant Key Agreement for Dynamic Collaborative Groups. 7th ACM Conference on Computer and Communication Security, 235-244 (2000)

Norranut Saguansakdiyotin received the B.S. degree from Chulachomklao Royal Military Academy, Thailand and M.S. degree in Computer Science from Rangsit University, Thailand. He also received the M.S. degree in Engineering (Software Engineering) from West Virginia University, USA. He has worked as a lecturer of Computer Engineering Department at Siam University,Thailand from 2002. His research interests are in computer networks and network security.

Pipat Hiranvanichakorn received the B.Eng.Hons. degree in Electrical Engineering from Chulalongkorn University, Thailand and M.E. and D.E. degrees in Information Processing from Tokyo Institute of Technology, Japan. He is currently working as Associate Professor in School of Applied Statistics, National Institute of Development Administration (NIDA). He has been involved in number of researches conducted in the department and also has supervised graduate students dissertation. His research areas are Cryptography and Network Security, Mobile Ad Hoc Networks.