A New Key Agreement Scheme Based on the Triple Decomposition Problem

Yeşem Kurt Peker
TSYS School of Computer Science, Columbus State University
4225 University Avenue Columbus, GA 31907
(E-mail: yesemk@gmail.com)
(Received Dec. 30, 2012; revised and accepted July 5, 2012)

Abstract

A new key agreement scheme based on the triple decomposition problem over non-commutative platforms is presented. A realization of the new scheme over braid groups is provided and the strengths of it over earlier systems that rely on similar decomposition problems are discussed. The new scheme improves over the earlier systems over braid groups by countering the linear algebra and length based attacks to the decomposition problem in braid groups.

Keywords: cryptographic protocols, key agreement schemes, non-commutative cryptography, public key cryptography

1 Introduction

One of the main tasks in cryptography is the encryption and decryption of private messages. A key exchange system allows the users to agree on a common key to encrypt and decrypt their messages. We call this common key the shared key. The secrecy of the shared key is the main concern in a key exchange system. The first key exchange scheme was introduced by Diffie and Hellman in 1976 [7], and independently by Merkle in 1978 [23]. The security of the Diffie-Hellman key exchange system relies on the difficulty of the Diffie-Hellman problem over finite fields. Emergence of index calculus attacks against the discrete logarithm problem, developments in quantum computing, and also curiosity of mathematicians have led to search for new cryptosystems; cryptosystems relying on hard problems of different natures.

Various key agreements schemes to work on commutative settings, enhancements on existing schemes, and their cryptanalysis have been studied since Diffie and Hellman proposed the concept of key agreement protocols (KAP) and public key cryptosystems (PKC) in 1976. In more recent years, the demand in mobile communications resulted in studies addressing security concerns in these communications. Authentication and key agreement protocols took their place as one of the pillars of security in these studies [16, 20, 25].

In 1999 Anshel et al. introduced a new key agreement scheme using non-commutative groups known as the Commutator KAP exploiting the difficulty of conjugacy search problem [1]. Following this new line, Ko et al. in 2000 and Cha et al. in 2001 proposed new schemes that also work over non-commutative groups [5, 18]. However these systems have been vulnerable to some attacks [6, 13, 14, 15, 21]. A more recent key agreement protocol that relies on solving multivariate equations on non-commutative rings has been proposed by Yagisawa in 2012 [28, 29].

In this paper we introduce a new way to do key exchange. The method works over non-commutative structures and is designed to counter the weaknesses of the earlier systems; in particular the weaknesses in the revised Diffie-Hellman like (DH-like) key exchange system [5]. The revised DH-like system is considered to be the stronger system among the earlier systems over non-commutative groups. It has been realized over braid groups and its security relies on the difficulty of the decomposition problem over braid groups. The decomposition problem is to decompose a given element u in the group, into $a_1 g a_2$ where g is known and a_1 and a_2 are unknowns that satisfy some commutativity conditions. Even though the decomposition problem over braid groups is hard in general, the special choices that had to be made for the system to be practical and the linear nature of the relations between the public and private keys allowed some attacks to yield feasible solutions [27].

The improvement in the new scheme is that the relation between the public and private keys is no longer only linear; one has to deal with quadratic relations in addition to linear ones in order to find a key that works. The underlying problem is again a type of decomposition problem which we call the "triple decomposition problem". Stated roughly, it is the problem of decomposing an element into parts in which there are three unknowns that satisfy certain commutativity and invertibility conditions.

There have been some attacks against decomposition problems over braid groups in general. One line of at-

tacks uses linear representations of braid groups. These have been successful against the revised DH-like system [6, 15, 21]. Another line of attack uses a length-based probabilistic approach to solve equations in certain cases over braid groups [12]. Also, an attack that uses Dehornoy's handle reduction method as a counter measure to diffusion and as a tool for simplifying braid words has been successful against the revised Diffie-Hellman like scheme [24]. We discuss the effectiveness of these attacks on the new system in section 6. A recent development in cryptanalyis of Commutator KAP and DH-like scheme is given in [26]. The polynomial-time algorithms they propose to solve the underlying problems in the aforementioned schemes do not seem to apply to the scheme proposed in this manuscript as is stated in [26] (page 18).

We start by describing the revised DH-like system in section 2. In section 3 we describe the new primitive and discuss the security for a general platform. We state some cases that need to be avoided so that the system truly relies on the triple decomposition problem.

In section 4 we give a rough exposition of braid groups; we list the properties that make them useful for cryptography. We also briefly mention the representations of braid groups as they are one of the tools in attacking the systems that uses braid groups as platforms.

In section 5 we describe the revised DH-like scheme in the braid group setting and concentrate on why certain types of attacks work against it.

In section 6 we discuss the new primitive over braid groups. We provide two settings for the new scheme over braid groups. The first setting is simple but exhibits a vulnerability. It is introduced in order to motivate the second setting where the relations between the private and the public keys are quadratic. This renders the linear representation attacks ineffective. Also, the subgroups in the final setting are generated by short generators and this foils length-based attacks. The choice of subsets also thwarts the attacks that use handle reduction. So, when known attacks against similar systems are considered, the new system stands strong.

The linear algebra attacks we consider are via Burau representation because the representation is simple and is more likely to yield feasible solutions.

In section 7 we discuss practicality of the new scheme in terms of key size and the complexity of the operations during key exchange. Another practical concern that requires further research is the authentication to be used with the key exchange. The protocol as described in the subsequent sections is secure against passive eavesdropping adversary, but it is not secure against active adversary; for example against man-in-the-middle attack. Secure protocols and properties required of such protocols for authenticated key exchange have been discussed in various articles such as [2, 3, 8, 9, 22].

Even though more research is required in order for the system to be promising in practice, we think it is a worthwhile direction to pursue as it brings a new perspective in non-commutative cryptography. Even if braid groups turn out to be not suitable as a platform, there may be other groups or monoids on which the new primitive can be explored and hopefully work.

In order to describe the scheme in a more general setting we take the underlying structure to be a monoid.

Definition 1. *A monoid is a set with an associative binary operation and an identity element. It is almost a group except that the elements may not be invertible.*

Notation 1. *Let G be a monoid. We write G multiplicatively and use the notation $[A, B] = 1$ for two subsets A, B of G when $ab = ba$ for all $a \in A$ and $b \in B$.*

Suppose Alice and Bob are the two parties who want to agree on a key.

2 DH-like Key Exchange

Let G be a non-commutative monoid and g an element in G. Let L and R be two subsets of G satisfying $[L, R] = 1$.

2.1 Setting the private and the public keys

1. Alice and Bob agree on who will use which subset, say Alice uses L and Bob uses R.

2. Alice randomly chooses $a_1, a_2 \in L$ and computes $u = a_1 g a_2$. Her private key is (a_1, a_2) and her public key is u.

3. Bob randomly chooses $b_1, b_2 \in R$ and computes $v = b_1 g b_2$. His private key is (b_1, b_2) and his public key is v.

2.2 Key exchange

To agree on a key Alice and Bob do the following:

1. Alice sends Bob her public key u.
2. Bob sends Alice his public key v.
3. Alice computes $a_1 v a_2 = a_1(b_1 g b_2)a_2$.
4. Bob computes $b_1 u b_2 = b_1(a_1 g a_2)b_2$.

Since a_i and b_i commute (because they come from L and R respectively), Alice and Bob agree on

$$\text{shared key} = a_1 b_1 g a_2 b_2.$$

2.3 Security

In genreal, for any key exchange system, one can proceed in one of the following ways to find the shared key:

1. Find the private key (or a pseudo-key, a key that works like a private key) of one of the users using his/her public key.

2. Find the shared key using the public keys of the users without necessarily finding a private key.

The first problem, in the Revised DH-like scheme, translates to finding a_1, a_2 in L given g and $u = a_1 g a_2$ in G. This is called the *decomposition problem.*

The second problem, in the Revised DH-like scheme, translates to finding $a_1 b_1 g b_2 a_2$ given $a_1 g a_2$ and $b_1 g b_2$. We call this the *Diffie-Hellman like composition problem.*

Depending on the particular platform and system parameters (the element g and the subsets L and R) there may be different approaches to solving these problems. In section 5 we will describe a realization of the scheme over braid groups and discuss attacks against this realization.

3 The New Scheme

3.1 The protocol

The system requires a non-commutative monoid G and two sequences containing 5 subsets of G each, say $\mathcal{A} = \{A_1, A_2, A_3, X_1, X_2\}$ and $\mathcal{B} = \{B_1, B_2, B_3, Y_1, Y_2\}$, satisfying the invertibility and commutativity conditions:

i. (Invertibility conditions) The elements of X_1, X_2, Y_1, Y_2 are invertible.

ii. (Commutativity conditions) $[A_2, Y_1] = 1$, $[A_3, Y_2] = 1$, $[B_1, X_1] = 1$, and $[B_2, X_2] = 1$.

3.1.1 Setting the private and the public keys

Suppose a monoid G and the subsets $\mathcal{A} = \{A_1, A_2, A_3, X_1, X_2\}$ and $\mathcal{B} = \{B_1, B_2, B_3, Y_1, Y_2\}$ satisfying i and ii above are fixed. Alice and Bob carry out the following steps:

1. Alice and Bob agree on who will use which set of subsets; say Alice uses $\mathcal{A}$ and Bob uses $\mathcal{B}$.

2. Alice randomly chooses $a_1 \in A_1$, $a_2 \in A_2$, $a_3 \in A_3$, $x_1 \in X_1$, $x_2 \in X_2$, and computes:

$$u = a_1 x_1, \quad v = {x_1}^{-1} a_2 x_2, \quad \text{and} \quad w = {x_2}^{-1} a_3.$$

Her private key is (a_1, a_2, a_3) and her public key is (u, v, w).

3. Bob randomly chooses $b_1 \in B_1$, $b_2 \in B_2$, $b_3 \in B_3$, $y_1 \in Y_1$, $y_2 \in Y_2$, and computes:

$$p = b_1 y_1, \quad q = {y_1}^{-1} b_2 y_2, \quad \text{and} \quad r = {y_2}^{-1} b_3.$$

His private key is (b_1, b_2, b_3) and his public key is (p, q, r).

3.1.2 Key exchange

To agree on a key Alice and Bob do the following:

1. Alice sends Bob her public key (u, v, w).

2. Bob sends Alice his public key (p, q, r).

3. Alice computes $a_1 p a_2 q a_3 r$.

4. Bob computes $u b_1 v b_2 w b_3$.

Note that Alice computes

$$\begin{aligned} a_1 p a_2 q a_3 r &= a_1 (b_1 \underline{y_1}) \underline{a_2} (y_1^{-1} b_2 \underline{y_2}) \underline{a_3} (y_2^{-1} b_3) \\ &= a_1 b_1 a_2 b_2 a_3 b_3 \end{aligned}$$

and Bob computes

$$\begin{aligned} u b_1 v b_2 w b_3 &= (a_1 \underline{x_1}) \underline{b_1} (x_1^{-1} a_2 \underline{x_2}) \underline{b_2} (x_2^{-1}) a_3 b_3 \\ &= a_1 b_1 a_2 b_2 a_3 b_3. \end{aligned}$$

The commuting conditions ensure that they agree on

$$\text{shared key} = \quad a_1 b_1 a_2 b_2 a_3 b_3.$$

The idea is to hide the private key by multiplying each component by some elements from the monoid. These elements are chosen from subsets satisfying the invertibility and commutativity conditions so that both parties compute the same key. Note that there are no conditions on the subsets A_1 and B_3. If security is not sacrificed, they may be chosen in a special way to make the system more practical; for instance to have smaller key sizes.

3.2 Security

For some choices of subsets in the scheme the shared key can be computed from the public keys immediately. If these cases are avoided it seems that the only way to attack the system is to deal with equations that relate the public and private keys, i.e. solve the system of equations

$$a_1 x_1 = u \tag{1}$$

$$x_1^{-1} a_2 x_2 = v \tag{2}$$

$$x_2^{-1} a_3 = w \tag{3}$$

for a_1, x_1, a_2, x_2, a_3 satisfying the invertibility and commutativity conditions. These conditions are automatically met if it is made sure that x_1, a_2, x_2, a_3 come from X_1, A_2, X_2, A_3 respectively. The problem is then to decompose v and w into elements from the respective subsets in such a way that the inverse of the last component of v is the first component of w. Note that because there are no restrictions on a_1, once x_1 is found, u can be decomposed into $a_1 x_1$ by taking $a_1 = u x_1^{-1}$.

Solving (2) requires v to be decomposed into three elements, and solving (3) requires w to be decomposed into two elements. The former is generally a harder task than the latter. One can think of (2) as a quadratic equation in terms of the unknowns simply by rewriting it in the form $a_2 x_2 = x_1 v$ whereas (3) is considered linear.

The main difference of the new scheme from the earlier systems over non-commutative structures is that one has to deal with quadratic relations which in terms of decomposition becomes the triple decomposition problem. In this setting the triple decomposition problem can be defined as:

Definition 2. *Let G be a non-commutative monoid and X, Y, A be subsets of G where elements of X and Y are invertible and satisfy some commutativity conditions. The Triple Decomposition Problem (TDP) in G is the problem of finding $x \in X, a \in A$, and $y \in Y$ given $u = xay \in G$.*

In the general setting, the security of the system relies on the triple decomposition problem. Depending on the specific platform and choices of subsets, the security may be stated more precisely. We would like to note that when we say solving the system we do not mean finding a private key in the underlying platform. Depending on the platform it may be possible to map the the system into a different structure where tools to deal with these types of equations exist. This is going to become clear when we discuss the system over a specific platform, namely braid groups.

In the following section we discuss the cases in which the shared key can be computed from the public keys immediately as well as the cases in which solving the system of equations does not require solving the triple decomposition problem. So, these cases should be avoided to have a system truly relying on the triple decomposition problem.

3.2.1 Cases to be avoided

In this section we give some cases in which the shared key can be computed without requiring to solve a quadratic system and hence should be avoided. These cases are in a way obvious cases that should be avoided over any platform. There may be other cases that needs to be avoided depending on the platform chosen. One should pay attention to such platform-specific cases.

The cases listed in the remarks 1 and 2 below allow immediate computation of the shared key from the public keys hence should be avoided.

Remark 1. *If $[X_1, Y_1] = 1$, $[X_2, Y_1] = 1$, and $[X_2, Y_2] = 1$ then the shared key can be computed from the public keys by*

$$(a_1x_1)(b_1y_1)(x_1^{-1}a_2x_2)(y_1^{-1}b_2y_2)(x_2a_3)(y_2b_3) = upvqwr.$$

Remark 2. *If $[A_2, B_1] = 1$, $[A_3, B_2] = 1$, and $[A_3, B_1] = 1$, then the shared key can be computed from the public keys by*

$$(a_1a_2a_3)(b_1b_2b_3) = uvwpqr.$$

Remark 3. *For the system to truly rely on solving* (2) *(i.e. quadratic equations) we need to make sure that there are many solutions to* (3).

When there are few solutions to (3), the security of the system mainly relies on (two instances of) decomposing an element into two elements: First decompose w into $x_2^{-1}a_3$, then substitute x_2^{-1} in (2) and decompose vx_2^{-1} into $x_1^{-1}a_2$.

Another case that needs to be necessarily avoided in order to have a true TDP involved is given in the following remark:

Remark 4. *If $[A_2, B_1] = 1$ and $[X_2, B_1] = 1$, or $[A_3, B_2] = 1$ and $[A_3, Y_1] = 1$, then the security of the system relies on the difficulty of decomposing an element into two elements.*

When $[A_2, B_1] = 1$, the shared key is $a_1a_2b_1b_2a_3b_3$. Multiplying the first two components of Alice's public key, we get the equation $a_1a_2x_2 = uv$. We need to check if having a_1a_2 together in the shared key and in the equation above implies any vulnerabilities. We need to investigate if/when decomposing uv into ax_2 with $a \in G$, $x_2 \in X_2$ suffices to compute the shared key. We need to check if

$$apqa_3r = \text{shared key}.$$

Substituting $pq = b_1b_2y_2$ on the left hand side above and then using $[A_3, Y_2] = 1$ and $a = uvx_2{}^{-1}$ for the second equality, and $a_3 = x_2w$ for the third equality on the left hand side we get

$$\begin{aligned} a(b_1b_2y_2)a_3y_2^{-1}b_3 &= (uvx_2{}^{-1})(b_1b_2)a_3b_3 \\ &= (uvx_2{}^{-1})(b_1b_2)x_2wb_3. \end{aligned} \tag{4}$$

Note that the shared key is $ub_1vb_2wb_3$. The last expression in equation (4) above is equal to the shared key if $[X_2, B_1] = 1$, because then we have $vb_1 = b_1v$. This explains the first case in Remark 4. Similar arguments apply for having b_1b_2 together and the case $[A_3, B_2] = 1$ and $[A_3, Y_1] = 1$.

4 Braid Groups

Braid groups are infinite, non-commutative, finitely presented groups with properties desirable in cryptography for practical applications [5, 10, 11]. They have been used in cryptosystems such as Arithmetica [1] and Diffie-Hellman like braid-based systems [5, 18]. Here we give a brief explanation of braid groups that helps to state these practical properties. We also very briefly mention two linear representations of braid groups that have been useful in attacking braid-based cryptosystems.

Definition 3. *The n-braid group $\mathcal{B}_n$ is an infinite, non-commutative (for $n \geq 3$) group defined by the following group presentation*

$$\mathcal{B}_n = \left\langle \sigma_1, \sigma_2, \ldots, \sigma_{n-1} \mid \begin{array}{ll} \sigma_i\sigma_j = \sigma_j\sigma_i, & |i-j| \geq 2 \\ \sigma_i\sigma_j\sigma_i = \sigma_j\sigma_i\sigma_j, & |i-j| = 1 \end{array} \right\rangle$$

The integer n is called the braid index *and the elements in $\mathcal{B}_n$ are called n-braids or simply (braid) words. The generators σ_i are called* Artin generators.

4.1 Normal Form and Computations with Braids

Definition 4. *The* fundamental braid of index n, *denoted by Δ, is defined to be Δ_{n-1} where Δ_i is defined inductively*

by:

$$\Delta_1 = \sigma_1$$
$$\Delta_i = \sigma_1 \cdots \sigma_i \Delta_{i-1}.$$

So, $\Delta = \sigma_1 \cdots \sigma_{n-1} \sigma_1 \cdots \sigma_{n-2} \ldots \sigma_1 \sigma_2 \sigma_1$.

Theorem 1. Δ^2 *generates the center of* $\mathcal{B}_n$ *for* $n \geq 3$.

Definition 5. *Let* $\mathcal{B}_n^+$ *denote the submonoid of* $\mathcal{B}_n$ *generated by* $\sigma_1, \ldots, \sigma_{n-1}$. *Elements in* $\mathcal{B}_n^+$ *are called* positive braids.

Definition 6. *There is a partial order on* $\mathcal{B}_n$ *defined by*

$$v \leq w \quad \textit{iff}$$

$$\textit{there exist } \alpha, \beta \in B_n^+ \quad \textit{such that} \quad w = \alpha v \beta.$$

Any $v \in \mathcal{B}_n$ *which satisfies* $\epsilon \leq v \leq \Delta$ *is called a* canonical factor *where* ϵ *is the empty word.*

There is a canonical homomorphism from the braid group $\mathcal{B}_n$ into the symmetric group S_n, say $\pi : \mathcal{B}_n \to S_n$, which maps σ_i onto the transposition interchanging i and $i+1$. The restriction of this homomorphism to the set of canonical factors induces a bijection. Due to this bijection a canonical factor is also called a *permutation braid.*

Definition 7. *Let* u *be a positive braid. A factorization* $u = vw$ *into a canonical factor* v *and a positive braid* w *is said to be* left weighted *if* v *has the maximal length among such decompositions.*

Definition 8. *Every braid word* W *in* $\mathcal{B}_n$ *can be written uniquely in the form* $\Delta^k p_1 p_2 \ldots p_l$ *where* Δ *is the fundamental braid of index* n, k *is an integer, and* p_i *are permutation braids satisfying* $\rho_i \rho_{i+1}$ *is left-weighted for* $1 \leq i \leq l-1$ *[10]. This form is called the* Garside normal form *and* l *is called the* canonical length *of* W.

The following are the properties that make braid groups suitable for applications.

1. The number of n-braids of canonical length l is at least $(\lfloor \frac{n-1}{2} \rfloor !)^l$ [18].
2. Let W be a word on σ_i's with word length s. Then the left-canonical form of W can be computed in time $O(s^2 \log n)$ [11]. This implies a fast solution to the word problem.
3. Each canonical factor can be represented by a permutation on n letters so requires $n \log n$ bits. Therefore a braid of canonical length l can be represented by a bit string of size $ln \log n$.
4. Let $U = \Delta^u p_1 \ldots p_l$ and $V = \Delta^v q_1 \ldots q_k$ be the left canonical forms of n-braids. Then the left canonical form of UV can be computed in time $O(lkn \log n)$ and the left canonical form of U^{-1} can be computed in time $O(ln)$ [11].

4.2 Linear Representations of Braids

Braid groups have two well-known linear representations: Burau Representation [4] and Lawrance-Krammer (LK) representation [19]. In the Burau representation, the braid group G_n is mapped into the matrix group $GL_{n-1}(\mathbb{Z}[t^{\pm 1}])$, the group of $(n-1) \times (n-1)$ matrices of Laurent polynomials over integers. The entries in the matrices in this representation are quite simple: Image of the generator σ_i is the $(n-1) \times (n-1)$ matrix obtained from the identity matrix by replacing the central $(i, i+1)$ square with $\left(\begin{smallmatrix} 1-t & t \\ 1 & 0 \end{smallmatrix}\right)$, i.e. image of σ_i is

$$\begin{bmatrix} I_{i-1} & 0 & 0 \\ 0 & \left(\begin{smallmatrix} 1-t & t \\ 1 & 0 \end{smallmatrix}\right) & 0 \\ 0 & 0 & I_{n-i-1} \end{bmatrix}$$

where I_k is the identity matrix of size k.

The small size matrices and simple entries make computations more efficient in Burau representation but the representation is not faithful for $n \geq 5$. On the other hand, Lawrence-Krammer representation is faithful but the matrices in this representation are big and the entries are not as simple. The braid group of index n is mapped into $n \times (n-1)$ matrices over Laurent polynomial ring in two variables over integers. In this paper we will give analysis of the system using Burau representation. Analysis via LK representation seems less efficient. Nevertheless it needs to be done. We hope to provide it in the very near future.

5 Revised DH-like Scheme Over Braids and Attacks Against It

The revised DH-like scheme requires two commuting subsets L and R (see section 2). In braid groups commutativity is easily achieved by taking generators apart from each other. In the system over braid groups L and R are taken to be the subgroups generated by the (almost) left half and the right half of the generators respectively [5]. i.e. $L = \langle \sigma_1 \ldots \sigma_{\lfloor \frac{n-1}{2} \rfloor} \rangle$ and $R = \langle \sigma_{\lfloor \frac{n-1}{2} \rfloor + 2} \ldots \sigma_{n-1} \rangle$.

Revised DH-like system in the above setting over braid groups has been vulnerable to various types of attacks. One type of attack is linear algebra attacks in which a representation of braid groups is used to map the system into a matrix group. The equations are solved in the matrix group for the (image of the) shared key. Once the (image of the) shared key is found, it is pulled back into the braid group. In order for this to work a faithful representation with a feasible algorithm to pull the solution back is required. Both Burau representation and LK representation have been used to attack the system. They have been effective against the revised DH-like system because of the combination of the properties of the system listed below. We consider the Burau representation here but attacks via LK representation make use of the same properties [6].

1. When the system is mapped into the matrices one gets a system of linear equations to solve. Recall that the equation relating the public and private keys is $a_1 g a_2 = u$. Rewriting this in the form

$$a_1 g = u a_2^{-1}$$

we get a linear equation.

Let $\rho : G_n \to GL_n(\mathbb{Z}[t^{\pm 1}])$ be the Burau representation of the braid group G_n and let A_1, G, A_2^{-1}, U be the images of a_1, g, a_2^{-1}, u under ρ. Then any solution (A_1', A_2') to the system $A_1 G = U A_2^{-1}$ satisfying the commuting condition that $A_i' M = M A_i'$ for all $M \in \rho(L)$ for $i = 1, 2$ allows the computation of (the image of) the shared key: If the image of Bob's public key is V then (image of) the shared key is $A_1' V A_2'$ because $A_1' V A_2' = A_1' B_1 G B_2 A_2' = B_1 A_1' G A_2' B_2 = B_1 U B_2$, which is how Bob would compute the shared key. Note that the system of equations is over Laurent polynomials with integer coefficients.

2. The braid words in the subsets L and R in the setting above are mapped to matrices in nice block forms so the system to solve is simpler than a random system. On Alice's side for example we have

$$A_1 = \begin{bmatrix} A_{11} & 0 \\ 0 & I \end{bmatrix}, \quad A_2 = \begin{bmatrix} A_{21} & 0 \\ 0 & I \end{bmatrix},$$

$$U = \begin{bmatrix} U_1 & U_2 \\ U_3 & U_4 \end{bmatrix}, \quad G = \begin{bmatrix} G_1 & G_2 \\ G_3 & G_4 \end{bmatrix}.$$

The entries are all $\frac{n}{2} \times \frac{n}{2}$ matrices (assuming n is even).

And so the system to solve is

$$\begin{bmatrix} A_{11}G_1 & A_{11}G_2 \\ G_3 & G_4 \end{bmatrix} = \begin{bmatrix} U_1 A_{21} & U_2 \\ U_3 A_{21} & U_4 \end{bmatrix}$$

3. The system can be reduced to positive braids [21]. In other words,

$$a_1 g a_2 = u \text{ with } a_1, a_2 \in L, g, u \in B_n \quad \text{iff}$$

$$a_1' g' a_2' = u' \text{ with } a_1', a_2' \in L^+, g', u' \in B_n^+$$

where L^+ consists of positive braid words generated by the first half of the generators. The reduced system gives matrices with polynomial entries instead of Laurent polynomials in the Burau representation. The computations are simpler when only polynomials are involved. Also, even though the Burau representation is not faithful, there is a heuristic algorithm to pull the matrix back into the the braid group when it has polynomial entries (and is in the image). The algorithm works with non-negligible probability.

Length-based approach for solving equations in braid groups and its applicability to the decomposition problem and the conjugacy search problem are discussed in [12, 13]. We will discuss length-based attacks in more detail in section 6.

Using Dehornoy's handle reduction method to minimize the lengths of the words has been another way of successfully attacking the DH-like system [24]. This special length-based method takes advantage of the nature of the equations to be solved as well. That the unknowns a_1, a_2 come from R_n (or L_n) and that the the middle component g in the equation $u = a_1 g a_2$ is known make this type of attack successful against the revised DH-like system.

6 New Primitive over Braids

A possible division of generators that satisfies the commutativity conditions and avoids the cases in remarks 1, 2, 4 in section 3.2 is given below.

Let G_n be the braid group of index n. Let $n - 1 = 3d$ for some positive integer $d \geq 2$. Set

$$\begin{array}{ll} A_1 = \mathcal{B}_n & \\ X_1 = \langle \sigma_1, \ldots, \sigma_{d-1} \rangle & B_1 = \langle \sigma_{d+1}, \ldots, \sigma_{n-1} \rangle \\ A_2 = \langle \sigma_1, \ldots, \sigma_{d-1} \rangle & Y_1 = \langle \sigma_{d+1}, \ldots, \sigma_{n-1} \rangle \\ X_2 = \langle \sigma_1, \ldots, \sigma_{2d-1} \rangle & B_2 = \langle \sigma_{2d+1}, \ldots, \sigma_{n-1} \rangle \\ A_3 = \langle \sigma_1, \ldots, \sigma_{2d-1} \rangle & Y_2 = \langle \sigma_{2d+1}, \ldots, \sigma_{n-1} \rangle \\ & B_3 = \mathcal{B}_n \end{array} \tag{5}$$

so that X_1 is generated by the first $d - 1$ generators and so on. The condition that the equation $x_2^{-1} a_3 = w$ has a large solution space (as was required in 3 in section 3.2) is satisfied in this setting, because we have $X_2 = A_3$ which gives $x_2 = x$, $a = xw$ is a solution for any $x \in X_2$. Even though the above choice of subsets takes the remarks in section 3.2 into consideration, the system is vulnerable to linear algebra attacks, in particular using Burau representation of braid groups.

The images of the braid words corresponding to the subgroups chosen in setting 5 are matrices in certain block forms. We assume $n - 1 = 3d$, so we can think of the matrices to be composed of 9 submatrices of size $d \times d$ each. We use the following convention for these submatrices: For example the matrices corresponding to braid words in A_1 and in X_1 are respectively of the form

$$a_1 = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{14} & A_{15} & A_{16} \\ A_{17} & A_{18} & A_{19} \end{bmatrix} \quad x_1 = \begin{bmatrix} X_{11} & 0 & 0 \\ 0 & I_d & 0 \\ 0 & 0 & I_d \end{bmatrix}.$$

Having matrices in block forms implies a vulnerability: some of the non-trivial entries in the images of private keys get revealed in the public keys when matrices are multiplied. (The entries of the submatrices that are 0 or I_d are called trivial) For example the first component of Alice's public key looks like

$$u = a_1 x_1 = \begin{bmatrix} A_{11}X_{11} & A_{12} & A_{13} \\ A_{14}X_{11} & A_{15} & A_{16} \\ A_{17}X_{11} & A_{18} & A_{19} \end{bmatrix}.$$

Notice that the second and third columns of u and a_1 are the same which means those entries of a_1 can be read off from the public key u. Another weakness in this setting is that the cases in remarks in section 3 are barely avoided. The shared key $a_1b_1a_2b_2a_3b_3$ is equal to $a_1a_2b_1a_3b_2b_3$ in this setting because a_2 and b_1, and a_3 and b_2 commute. This excess commutativity together with the revealed entries weakens the system. We modify the subsets in setting 6 to counter these weaknesses. The modification is based on the simple observation that if $ab = ba$, then $(sas^{-1})(sbs^{-1}) = (sbs^{-1})(sas^{-1})$ for any $s \in G_n$.

Let $s_1, s_2, s_3, s_4 \in G_n$ be fixed, system wide parameters and let X_1, X_2, A_1, A_2, A_3, Y_1, Y_2, B_1, B_2, B_3 be as in 5. Set

$$\begin{aligned} A'_1 &= G_n & & (6)\\ X'_1 &= \{s_1 x s_1^{-1} \mid x \in X_1\} & B'_1 &= \{s_1 b s_1^{-1} \mid b \in B_1\}\\ A'_2 &= \{s_2 a s_2^{-1} \mid a \in A_2\} & Y'_1 &= \{s_2 y s_2^{-1} \mid y \in Y_1\}\\ X'_2 &= \{s_3 x s_3^{-1} \mid x \in X_2\} & B'_2 &= \{s_3 b s_3^{-1} \mid b \in B_2\}\\ A'_3 &= \{s_4 a s_4^{-1} \mid a \in A_3\} & Y'_2 &= \{s_4 y s_4^{-1} \mid y \in Y_2\}\\ & & B'_3 &= G_n. \end{aligned}$$

The commutativity conditions are still satisfied with these subsets. With careful choice of s_i's (by careful we mean s_i should include a large number of distinct generators) the block forms are destroyed so no entry gets revealed in the public keys. The other weakness in the previous setting, namely the commutativity of certain parts of the shared key is also avoided. The only case in the remarks in section 3.2 that is not immediately taken care of is Remark 3, namely the condition that equation $x_2'^{-1}a'_3 = w$ has a large solution space for $x'_2 \in X'_2$ and $a'_3 \in A'_3$. Written more explicitly the equation

$$s_3 x_2^{-1} s_3^{-1} s_4 a_3 s_4^{-1} = w$$

or equivalently

$$x_2^{-1} s_3^{-1} s_4 a_3 = s_3^{-1} w s_4 = w' \quad (7)$$

should have a large solution space for $x_2 \in X_2$ and $a_3 \in A_3$. We will discuss this issue below. Before, we would like to draw attention to the fact that this equation is similar to the equation that needs to be solved in the revised DH-like KE system over braid groups [5]. Namely, find $x, y \in H \subset G$ given $u = xay$ and $a \in G$. In the setting of the Dh-like scheme over braid groups, H is generated by either the left or the right half of the generators. The case in the new system is different in two ways:

1. H corresponds to X_2 in our case and it consists of a larger potion of the generators(two thirds instead of a half).

2. A solution to equation 7 does not necessarily lead to a working private key. The variable x_2 has to also satisfy

$$s_1 x_1^{-1} s_1^{-1} s_2 a_2 s_2^{-1} s_3 x_2 s_3^{-1} = v$$

or equivalently

$$x_1^{-1} s_1^{-1} s_2 a_2 s_2^{-1} s_3 x_2 = s_1^{-1} v s_3 = v'. \quad (8)$$

The first point is worth mentioning because the techniques that were able to get solutions for parameters suggested in [18] made use of the special structure of the subgroup H or more accurately the image of H under linear representations of G_n (see section 5 and [6, 21]). When we have a larger H this structure changes and the same techniques may not work.

More important is the second point. One way to proceed is to substitute a solution of (7) into (8). This gives another equation of similar type. If this new equation has a solution that we can compute then we have a key, if not we try another solution of (7). This works if the solution space for (7) is small. Otherwise one has to deal with (8) as a whole which involves quadratic relations.

Now the question is how to make sure that (7) has a large solution space. We need the definition of a centralizer before we proceed:

Definition 9. *Let G be a monoid (or a group) and let $g \in G$. The centralizer of g in G is the set of all elements in G that commutes with g and is denoted by $C(g)$. i.e.*

$$C(g) = \{a \in A | ag = ga\}.$$

Note that, in the above setting, if $x_2^{-1} = x$, $a_3 = a$ is a solution to the system, then so is $x_2^{-1} = xz$, $a_3 = z^{-1}a$ for any $z \in C(s_3^{-1}s_4) \cap X_2$. (Recall that in setting (5) the subgroups X_2 and A_3 are equal so we have $xz \in X_2$ and $z^{-1}a \in A_3$ as required.) Therefore in order to guarantee a large solution space for (7) we can choose s_3 and s_4 so that $C(s_3^{-1}s_4) \cap X_2$ is large. Similarly, for the corresponding party, we require $C(s_1^{-1}s_2) \cap Y_1$ to be large. Here we will discuss the case for equations on Alice's side. The conclusions for Bob's side can be derived similarly. One way of having many elements in $C(s_3^{-1}s_4) \cap X_2$ is for $C(s_3^{-1}s_4)$ and X_2 to have several common generators. Recall that X_2 is generated by $\sigma_1, \ldots, \sigma_{2d-1}$. For example, if s_3 and s_4 are chosen so that $s_3^{-1}s_4$ consists of generators from $\sigma_{d+1}, \ldots, \sigma_{n-1}$, then X_2 and $C(s_3^{-1}s_4)$ will have $\sigma_1, \ldots, \sigma_{d-1}$in common so (7) will have many solutions. (On Bob's side s_1 and s_2 would be chosen so that $s_1^{-1}s_2$ consist of generators from $\sigma_1, \ldots, \sigma_{2d-1}$).

So, rewriting (8), the equation to solve in order to find a pseudo-key becomes:

Problem: Given $v, s, s' \in G_n$ decompose v into $x_1^{-1} s a_2 s' x_2$ where $x_1 \in X_1$, $a_2 \in A_2$, and $x_2 \in X_2$.

This problem is again a decomposition problem and it involves decomposing an element into five pieces three of which are unknown. So it is a variant of the triple decomposition problem and involves quadratic equations. There is no known algorithm to solve this problem. There are

two lines of attacks against decomposition problems over braid groups in general: Linear algebra attacks and length based attacks. As we have explained in section 5.1 linear algebra attacks have been effective against the decomposition problem in the revised DH-like scheme. Length based attacks have been effective against the conjugacy search problem in the DH-like cryptosystem in [18], and the multiple conjugacy search problem in the commutator key exchange protocol in [1]. In the next two sections we discuss the effectiveness of these methods against the new system.

6.1 Linear algebra attacks

We give an analysis via Burau representation. We consider the equations on Alice's side. Let the images of the braid words in (7) and (8) under Burau representation be as follows. Note here that the entries of $s_1^{-1}s_2, s_2^{-1}s_3, s_3^{-1}s_4$, and $s_2^{-1}s_4$ are known and the rest are unknown.

$$s_1^{-1}s_2 = \begin{bmatrix} s1 & s2 & 0 \\ s3 & s4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad s_2^{-1}s_3 = \begin{bmatrix} r1 & r2 & r3 \\ r4 & r5 & r6 \\ r7 & r8 & r9 \end{bmatrix}$$

$$s_3^{-1}s_4 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & t1 & t2 \\ 0 & t3 & t4 \end{bmatrix} \qquad s_2^{-1}s_4 = \begin{bmatrix} p1 & p2 & p3 \\ p4 & p5 & p5 \\ p7 & p8 & p9 \end{bmatrix}$$

$$x_1^{-1} = \begin{bmatrix} x11 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad a_2 = \begin{bmatrix} a21 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$x_2 = \begin{bmatrix} x21 & x22 & 0 \\ x23 & x24 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad x_2^{-1} = \begin{bmatrix} ix21 & ix22 & 0 \\ ix23 & ix24 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$a_3 = \begin{bmatrix} a31 & a32 & 0 \\ a33 & a34 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

From (7) we get w':

$$w' = \begin{bmatrix} ix21\ a31 + ix22\ t1\ a33 & ix21\ a32 + ix22\ t1\ a34 & ix22\ t2 \\ ix23\ a31 + ix24\ t1\ a33 & ix23\ a32 + ix24\ t1\ a34 & ix24\ t2 \\ t3\ a33 & t3\ a34 & t4 \end{bmatrix}$$

and from (8) we get v' :

First column of $v' =$

$$\begin{bmatrix} (x11\ s1\ a21\ r1 + x11\ s2\ r4)\ x21 + (x11\ s1\ a21\ r2 + x11\ s2\ r5)\ x23 \\ (s3\ a21\ r1 + s4\ r4)\ x21 + (s3\ a21\ r2 + s4\ r5)\ x23 \\ r7\ x21 + r8\ x23 \end{bmatrix}$$

Second column of $v' =$

$$\begin{bmatrix} (x11\ s1\ a21\ r1 + x11\ s2\ r4)\ x22 + (x11\ s1\ a21\ r2 + x11\ s2\ r5)\ x24 \\ (s3\ a21\ r1 + s4\ r4)\ x22 + (s3\ a21\ r2 + s4\ r5)\ x24 \\ r7\ x22 + r8\ x24 \end{bmatrix}$$

Third column of $v' =$

$$\begin{bmatrix} x11\ s1\ a21\ r3 + x11\ s2\ r6 \\ s3\ a21\ r3 + s4\ r6 \\ r9 \end{bmatrix}$$

Since v' and w' are public, we can multiply them to get $v'w'$ which yields more equations:

First column of $v'w' =$

$$\begin{bmatrix} (x11\ s1\ a21\ p1 + x11\ s2\ p4)\ a31 + (x11\ s1\ a21\ p2 + x11\ s2\ p5)\ a33 \\ (s3\ a21\ p1 + s4\ p4)\ a31 + (s3\ a21\ p2 + s4\ p5)\ a33 \\ p7\ a31 + p8\ a33 \end{bmatrix}$$

Second column of $v'w' =$:

$$\begin{bmatrix} (x11\ s1\ a21\ p1 + x11\ s2\ p4)\ a32 + (x11\ s1\ a21\ p2 + x11\ s2\ p5)\ a34 \\ (s3\ a21\ p1 + s4\ p4)\ a32 + (s3\ a21\ p2 + s4\ p5)\ a34 \\ p7\ a32 + p8\ a34 \end{bmatrix}$$

Third column of $v'w' =$:

$$\begin{bmatrix} x11\ s1\ a21\ p3 + x11\ s2\ p5 \\ s3\ a21\ p3 + s4\ p5 \\ p9 \end{bmatrix}$$

Note that the discussions in the previous paragraph are taken into account so that $s_3^{-1}s_4$ and $s_1^{-1}s_2$ have the necessary block forms. Also, not only the components of public information v', w', but also the matrix corresponding to $v'w'$ is included as it may lead to useful equations. In the entries of v',w', and $v'w'$ all the variables are $d \times d$ matrices where $d = \frac{n-1}{3}$. The single- indexed variables are known; they are system parameters, the double-indexed ones are unknown. Looking at the entries of v', w', and $v'w'$ (each of which is an equation) we draw attention to the following points:

1. There are linear and quadratic equations. Not all solutions to the linear equations lead to the shared key; the solution has to satisfy a quadratic equation as well. In fact, the choice of subsets ensures that there are too many solutions to the linear equations to exhaust by trial and error.

2. The unknowns are again in nice block forms but the resulting equations are not as simple as they are in the revised DH-like system (see section 5)

3. The system cannot be reduced to positive braids. Or more accurately when the system is reduced to positive braids the respective subsets are not preserved in equation $x_1^{-1}a_2x_2 = v$ so that the nice block forms that allow one to easily solve the equation are destroyed.

   The idea when reducing the system to positive braids is to multiply the equation by high enough powers of the "appropriate fundamental braids" so that the unknowns when these fundamental braids are included are positive. "Appropriate fundamental braid" is obtained in a similar fashion as in Definition 4 by using the generators that generate the respective subgroup. This makes sense in this setting because successive generators are used to generate each subgroup. In equation $x_1^{-1}a_2x_2 = v$, when a power of the "fundamental braid" in the subgroup A_2 is multiplied (on the left or on the right) it can't be moved next to a_2 to put the system in the form $x_1'^{-1}a_2'x_2' = v'$ with $x_1' \in X_1^+, a_2 \in A_2^+, x_2 \in X_2^+$ because "fundamental braid" in the subgroup A_2 does not necessarily

commute with x_1 or x_2. This method works in the earlier system, because the middle element in the system comes from B_n. Since Δ^2 commutes with every word in the group (Theorem 1), multiplying the system by a high enough even power of Δ and moving it into the middle using commutativity, one is able to get a positive braid in B_n^+ in the middle as desired.

6.2 Length-based attacks

In [12] Garber et al give a probabilistic method to solve a system of equations in a random finitely generated subgroup of the braid group. They make use of a "monotonic" length function - a length function which satisfies that the expected length tends to increase with the number of generators (of the subgroup) multiplied. In the analysis they provide, the subgroup is generated by elements that are composed of 10 Artin generators each. They state that if the generators can be written as a product of very few Artin generators then the required monotonicity of the length function gets violated and the algorithm fails. According to the setting in (6)and the discussion following it, the generators of the subgroups to which the unknowns belong are single Artin generators in the new system. Most of the analysis provided in [12] is over $\mathcal{B}_8$, braid group of index 8. For larger index, it is concluded that the probability of success decreases but not significantly. The discussion in this paper seemed insufficient to make conclusions for the new system which will be using a higher index (100), so we referred to another paper by the same authors [13] where the multiple conjugacy search problem instead of the general decomposition problem is considered. The method applies more effectively to the conjugacy problem because of the special nature of the equations involved: $x^{-1}ax = u$ where x is the unknown.) Some of the conclusions they have apply to any decomposition problem. First conclusion is the same as in [12], that is, the smaller is the size of the generators of the subgroups, the smaller is the probability of success. Here size of an element is the number of Artin generators in it. Since the generators consist of single Artin generators in the new system, the system seems to be strong against length-based attacks. Another conclusion is that the longer is the size of the unknowns, the smaller is the probability of success. The analysis in [13] was carried over B_n for n up to 20 and for unknowns of length up to 18 in the generators of the subset to which they belong.

These points need to be considered when the parameters to be used in practice are chosen. A plausible choice of parameters is to use the braid group of order 100, i.e. $\mathcal{B}_{100}$, and choose the secret values to consist of about 300 generators each. In the setting in 6, $\mathcal{B}_{100}$ gives subgroups of 32 generators and 64 generators. Each of these generators of the subgroups is a single Artin generator and this foils the length-based attacks.

Another length-based attack to solve the decomposition problem of finding $x_1, x_2 \in R_n$ given $a \in \mathcal{B}_n$ and $u = x_1 a x_2$ is given in [24]. It uses Dehornoy's handle reduction to simplify the braid words. The idea is to counter-measure the diffusion provided by the Garside normal by converting the word u (which is given in its Garside normal form) to $x_1' a x_2'$ with $x_1', x_2' \in R_n$ and a of short length. This attack takes advantage of the fact that the middle component in the decomposition of u is known and the fact that x_1 and x_2 come from R_n (or L_n).

The equations we need to solve in the new system, namely equations (7) and (8), are different from the equations attacked using this method. First, the middle component in (8), a_2, is unknown. Second, even though the middle component in (7) is known, the unknowns do not come from L_n or R_n but from larger subsets. Also, it is made sure while choosing the system parameters that not all solutions to (7) will provide a solution for (8) (See beginning of section 6).

7 Key Size and Complexity of Key Computation

In section 3.1, we discussed the complexity of some basic operations of braids when represented in Garside normal form. Garside normal form is not the only way to uniquely represent braid words. How to choose a random braid word, which canonical form of the braids to use for security and practicality is an interesting question in itself and there have been studies addressing these in the literature [5, 17].

Assuming braid words are represented in Garside normal form and there is a good way of choosing the words randomly, the key size for a private key, say Alice's, would be $3ln \log n$ where l is the canonical length, n is the braid index. This is because we have three braid words in the private key (a_1, a_2, a_3) and Property 3 on page 5 in section 3.1 states that the number of bits for a braid word of canonical length l is $ln \log n$. Same is true for the public key, however we note that l should be taken to be the maximum possible length among lengths of all private and public keys.

The computation of the shared key requires 5 braid operations (for example $a_1 p a_2 q a_3 r$ when Alice computes the shared key). Using Property 4 in section 3.1 and assuming l is the largest canonical length that can come up in these operations, the complexity of calculating the shared key in terms of system parameters is $5l^2 n \log n$.

Note that, even though high for the parameters suggested in section 5.2 compared to systems used in practice, the complexity of key calculation is polynomial in the number of bits in the private key assuming l is constant (or polynomial in n); i.e if $K = 3l \log n$ is the key size, then computation of the shared key takes $\frac{5}{3} lK$ operations.

More research need to be conducted in order to determine how the canonical length can be kept below a certain value for practical purposes.

8 Conclusion

We have proposed a new way to achieve key agreement in a public key system. The security of the new scheme relies on what we call the triple decomposition problem in a non-commutative group; namely decomposing an element into various pieces three of which are unknown and satisfy certain invertibility and commutativity conditions. We have focused on braid groups as they have the desirable practical properties required by the system. We analyzed the system over a classical protocol in detail and provided a setting in which the system is immune to linear algebra attacks via Burau representation and length-based attacks. Further research is required to establish a stronger confidence in the system and to determine concrete parameters for practical purposes. Concerns in determining parameters for braid based applications such as how to generate random braids are partly addressed in [17].

Another enhancement to the system is to employ a new protocol suggested by Shpilrain and Ushakov in [27]. In this protocol, instead of fixing commuting subsets in advance, the commuting conditions are satisfied during key exchange by exchanging centralizers of random elements. The underlying problem becomes finding the common centralizer of a set of elements. This is another promising direction that we would like to do further research on.

References

[1] I. Anshel, M. Anshel, B. Fisher, and D. Goldfeld, "New key agreement protocols in braid group cryptography," in *Topics in Cryptology - CT-RSA 2001, LNCS 2020*, pp. 13–27, 2001.

[2] M. Bellare, D. Pointcheval, and P. Rogaway, "Authenticated key exchange secure against dictionary attacks," in *Advances in Cryptology – Eurocrypt 2000, LNCS 1807*, pp. 139–139, 2000.

[3] S.M. Bellovin and M. Merritt, "Encrypted key exchange: Password-based protocols secure against dictionary attacks," in *Proceedings of the I.E.E.E. Symposium on Research in Security and Privacy*, pp. 72–72, Oakland, May 1992.

[4] W. Burau, "Uber zopfgruppen und gleichsinning verdrillte verkettungen," *Abh. Math. Sem. Ham. II*, pp. 171–178, 1936.

[5] J. Cha, K. Ko, S. Lee, J.Han, and J. Cheon, "An efficient implementation of braid groups," in *Proc. of Asiacrypt 2001, LCNS 2248*, pp. 1–13, Oakland, 2012.

[6] J. Cheon and B. Jun, "A polynomial time algorithm for the braid-diffie-hellman conjugacy problem," in *Proc. of Crypto 2003, LNCS 2729*, pp. 212–225, 2003.

[7] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory, IT-22*, vol. 6, pp. 654–664, 1976.

[8] W. Diffie, P.C. van Oorschot, and M.J. Wiener, "Authentication and authenticated key exchange," *Design, Codes, and Cryptography*, vol. 2, pp. 107–125, 1992.

[9] E.J.Lu, C.C. Lee, and M.S. Hwang, "Cryptanalysis of some authenticated key agreement protocols," *International Journal of Computational and Numerical Analysis and Applications*, vol. 3, no. 2, pp. 151–157, 2003.

[10] E.A. Elrifai and H.R. Morton, "Algorithms for positive braids," *Quart. J. Math*, vol. 45, pp. 479–497, 1994.

[11] D. Epstein, J. Cannon, D. Holt, S. Levy, M. Paterson, and W. Thurston, *Word Processing in Groups*. Jones & Barlett, 1992.

[12] D. Garber, S. Kaplan, M. Teicher, B. Tsaban, and U. Vishne, "Probabilistic solutions of equations in the braid group," *Advances in Applied Mathematics*, vol. 35, pp. 323–334, 2005.

[13] D. Garber, S. Kaplan, M. Teicher, B. Tsaban, and U. Vishne, "Length-based conjugacy search in the braid group," *Contemporary Mathematics*, vol. 418, pp. 75–87, 2006.

[14] D. Hofheinz and R. Steinwandt, "A practical attack on some braid group based cryptoraphic primitives," in *Public Key Crypography-PKC 2003, LNCS 2567*, pp. 187–198, 2003.

[15] J. Hughes, "A linear algebraic attack on the AAFG1 braid group cryptosystem," in *7th Australasian Conference on Information Security and Privacy -ACISP 2002, LNCS 2384*, pp. 176–189, 2002.

[16] M.S. Hwang, S.K. Chong, and H.H. Ou, "On the security of an enhanced UMTS authentication and key agreement protocol," *European Transactions on Telecommunications*, vol. 22, no. 3, pp. 99–112, 2011.

[17] K. Ko, J. Lee, and T. Thomas, "Towards generating secure keys for braid cryptography," *Design, Codes, and Cryptography*, vol. 45, no. 3, pp. 317–333, 2007.

[18] K. Ko, S. Lee, J. Cheon, J. Han, J. Kang, and C. Park, "Public key cryptosystem using braid groups," in *Proc. of Crypto 2000, LNCS 1880*, pp. 166–183, 2000.

[19] D. Krammer, "Braid groups are linear," *Annals of Mathematics*, vol. 155, pp. 131–156, 2002.

[20] C.C. Lee, M.S. Hwang, and T.C. Lin, "A key agreement scheme for satellite communications," *Information Technology and Control*, vol. 39, no. 1, pp. 43–47, 2010.

[21] E. Lee and J. Park, "Cryptanalysis of the public key encryption braid groups," in *Proc. of Eurocrypt 2003, LNCS 2656*, pp. 477–490, 2003.

[22] R. Lu and Z. Cao, "Off-line password guessing attack on an efficient key agreement protocol for secure authentication," *International Journal of Network Security*, vol. 3, no. 1, pp. 35–38, 2006.

[23] R. Merkle, "Secure communications over insecure channels," *Communications of the ACM*, vol. 21, pp. 294–299, 1978.

[24] A. Myasnikov, V. Shpilrain, and A. Ushakov, "A practical attack on a braid group based cryptographic protocol," *Advances in Cryptology, LNCS 3621*, vol. 3621, 2005.

[25] H.H. Ou, I.C. Lin, and M.S. Hwang, "An effective AKA protocol for UMTS," *International Journal of Mobile Communciations*, vol. 10, no. 4, pp. 427–448, 2012.

[26] B. Tsaban, "Polynomial time solutions of computational problems in noncommutative-algebraic cryptograpy," *IACR Cryptology ePrint Archive 2012*, vol. 615, 2012.

[27] V.Shilparin and A. Ushakov, "A new key exchange protocol based on the decomposition problem," *Contemp. Math., Amer. Math. Soc.*, vol. 418, pp. 161–167, 2006.

[28] M. Yagisawa, "Key agreement protocols using multivariate quadratic equations on non-commutative ring," *IACR Cryptology ePrint Archive 2010*, vol. 458, 2010.

[29] M. Yagisawa, "Key distribution system and attribute-based encryption," *IACR Cryptology ePrint Archive 2012*, vol. 241, 2012.

Yeşem Kurt Peker is an assistant professor in the TSYS School of Computer Science at Columbus State University in Columbus, GA in the U.S. She received her B.S. degrees in Computer Engineering and Mathematics from Middle East Technical University (METU) in Ankara, Turkey. She received her M.S. degree in Mathematics from METU in Ankara, Turkey and Ph.D. from Indiana University-Bloomington in Bloomington, IN in the U.S. After receiving her Ph.D., she taught Mathematics at liberal arts colleges before she started her current position. Her research includes information and network security with a focus on cryptography. She is particularly interested in developing new cryptographic schemes with different underlying structures.