Des. Codes Cryptogr. (2007) 45:317–333
DOI 10.1007/s10623-007-9123-0

Towards generating secure keys for braid cryptography

Ki Hyoung Ko · Jang Won Lee · Tony Thomas

Received: 30 April 2007 / Revised: 6 August 2007 / Accepted: 6 August 2007 /
Published online: 8 September 2007

Abstract Braid cryptosystem was proposed in CRYPTO 2000 as an alternate public-key cryptosystem. The security of this system is based upon the conjugacy problem in braid groups. Since then, there have been several attempts to break the braid cryptosystem by solving the conjugacy problem in braid groups. In this article, we first survey all the major attacks on the braid cryptosystem and conclude that the attacks were successful because the current ways of random key generation almost always result in weaker instances of the conjugacy problem. We then propose several alternate ways of generating hard instances of the conjugacy problem for use braid cryptography.

Keywords Braid groups · Conjugacy problem · Braid cryptography

AMS Classifications 94A60 · 20F36 · 20F10

1 Introduction

The braid group cryptography was born with the pioneering work of Anshel et al. in 1999 [1] and Ko et al. in 2000 [20]. Since then, the braid groups have fascinated many cryptologists with its hard problems and efficient algorithms for parameter generation and group operations [5]. In particular, the conjugacy problem first introduced in the 1920s, with no

Communicated by P. Wild.

K. H. Ko (✉) · J. W. Lee · T. Thomas
Department of Mathematics, Korea Advanced Institute of Science and Technology, Daejeon 305-701, Korea
e-mail: knot@knot.kaist.ac.kr

J. W. Lee
e-mail: leejw@knot.kaist.ac.kr

T. Thomas
e-mail: thomas@knot.kaist.ac.kr

polynomial-time algorithm known till date (for braid groups with index $n \geq 5$), have been used in designing both public-key encryption schemes above and many other public-key cryptographic protocols [7,21,27,28]. This has then accelerated the attempts to solve the conjugacy problem in braid groups to break the security of the braid cryptosystems.

In this article, we first survey and analyze all the well known solutions to the conjugacy and related problems in braid groups and conclude that the attacks on braid cryptosystems were successful because the current ways of random key generation almost always result in weaker instances of the conjugacy problem. We then propose several ways of generating hard instances of the conjugacy problem for use braid cryptography. In Sect. 2, we briefly describe the braid groups, braid cryptosystems and survey all the well known solutions to the conjugacy and related problems in braid groups. In Sect. 3, we analyze the current ways of random key generation for braid cryptography. In Sect. 4, we propose several ways for the generation of secure keys for braid cryptography. The paper concludes with some remarks in Sect. 5.

2 Braid groups and the conjugacy problem

In this section, we give a brief review of the braid groups, braid cryptography and known solutions to the conjugacy and related problems. A good introduction to the braid groups is [2] and survey articles to braid cryptography are [7,21].

2.1 Braid groups

A braid is obtained by laying down a number of parallel strands and intertwining them so that they run in the same direction. The number of strands is called the braid index. Braids have the following geometric interpretation: an n-braid where $(n > 0)$ is a set of disjoint n strands all of which are attached to two horizontal bars at the top and bottom such that each strand always heads downwards as one moves along the strand from top to bottom. Two braids are equivalent if one can be deformed to the other continuously in the set of braids.

Let B_n be the set of all n-braids. B_n has a natural group structure. Each B_n is an infinite torsion-free noncommutative group and its elements are called n-braids. The multiplication ab of two braids a and b is the braid obtained by positioning a on the top of b. The identity e is the braid consisting of n straight vertical strands and the inverse of a is the reflection of a with respect to a horizontal line.

Any braid can be decomposed as a product of simple braids. One type of simple braids are the *Artin generators* σ_i, that have a single crossing between the i^{th} strand and the $(i+1)^{th}$ strand with the convention that the i^{th} strand crosses under the $(i+1)^{th}$ strand.

For each integer $n \geq 2$, the n-braid group B_n has the Artin presentation by generators $\sigma_1, \sigma_2, \ldots, \sigma_{n-1}$ with relations,

$$\begin{aligned} \sigma_i\sigma_j &= \sigma_j\sigma_i, \quad \text{where} \quad |i-j| \geq 2, \quad \text{and} \\ \sigma_i\sigma_{i+1}\sigma_i &= \sigma_{i+1}\sigma_i\sigma_{i+1}, \quad \text{for} \quad 1 \leq i \leq n-2. \end{aligned}$$

Let Σ_n be the symmetric group on $\{1, \ldots, n\}$. There is a natural homomorphism, $p : B_n \to \Sigma_n$, under which the generator σ_i of B_n maps to the transposition $(i, i+1)$ of Σ_n.

By B_n^+, we denote the submonoid of B_n generated by $\{\sigma_1, \ldots, \sigma_{n-1}\}$. Elements of B_n^+ are called the *positive braids*. A positive braid is characterized by the fact that at each crossing the strand going from left to right undercrosses the strand going from right to left.

A partial order $\prec$ on B_n^+ is given by saying that $x \prec y$ for $x, y \in B_n^+$, if x is a (left) *subword* of y, that is, $xz = y$ for some $z \in B_n^+$. Given $x, y \in B_n^+$, the (left) *join* $x \vee y$ of x and y is the minimal element with respect to $\prec$ among all z's satisfying that $x \prec z$ and $y \prec z$, and the (left) *meet* $x \wedge y$ of x and y is the maximal element with respect to $\prec$ among all z's satisfying that $z \prec x$ and $z \prec y$. One can similarly define the *right join* and *right meet*.

The *fundamental braid* $\Delta = (\sigma_1 \ldots \sigma_{n-1})(\sigma_1 \ldots \sigma_{n-2}) \ldots (\sigma_1\sigma_2)\sigma_1$ plays an important role in the study of B_n. Since, it represents a half twist as a geometric braid, $x\Delta = \Delta\tau(x)$ for any braid where τ denotes the involution of B_n sending σ_i to σ_{n-i}. It also has the property that $\sigma_i \prec \Delta$ for each $i = 1, \ldots, n-1$. For $S_n = \{x \in B_n^+ \mid x \prec \Delta\}$, the restriction $p : S_n \to \Sigma_n$ becomes a $1:1$ correspondence and an element in S_n is called a permutation braid.

The product ab of a permutation braid a and a positive braid b is *left weighted* if $(a^{-1}\Delta) \wedge b = e$, where e denotes the trivial braid (empty word). Each braid $x \in B_n$ can be uniquely written as,

$$x = \Delta^u x_1 x_2 \ldots x_k$$

where for each $i = 1, \ldots, k$, $x_i \in S_n - \{e, \Delta\}$ and $x_i x_{i+1}$ is left weighted. This decomposition is called the *left weighted form* of x. The weighted form provides a solution to the word problem in B_n and the integers u, $u + k$ and k are well-defined and are called the *infimum*, *supremum* and the *canonical length* of x, denoted by $\inf(x)$, $\sup(x)$ and $\ell(x)$ respectively.

2.2 Conjugacy problem and mathematical solutions

The conjugacy problem asks whether given two braids in B_n are conjugate each other. The first algorithmic solution to the conjugacy problem was due to Garside [11]. His solution and all other mathematical solutions use some finite invariant subsets of conjugacy classes as follows:

For $x \in B_n$, let $C(x) = \{a^{-1}xa | a \in B_n\}$ denote the conjugacy class of x. And let $I(x)$ be a subset of $C(x)$ satisfying the following properties:

(1) For any $x \in B_n$, the set $I(x) \subset C(x)$ is finite, non-empty and only depends on the conjugacy class of x. Two elements $x, y \in B_n$ are conjugate if and only if $I(x) = I(y)$, or, equivalently, if and only if $I(x) \cap I(y) \neq \emptyset$.
(2) Given any non-empty subset I of $I(x)$, there is a finite process which either proves that $I = I(x)$ or produces an element $z \in I$ and an element $c \in B_n$ such that $c^{-1}zc \in I(x) \backslash I$.

Thus, given any $x, y \in B_n$, the solution to the conjugacy problem involves the following steps:

(1) Find representatives $\tilde{x} \in I(x)$ and $\tilde{y} \in I(y)$;
(2) Repeatedly use the process (2) from above to construct new elements of $I(x)$ until $\tilde{y}$ is found as an element of $I(x)$.

2.2.1 Invariant subsets of a conjugacy class

Let $\inf_c(x)$ and $\sup_c(x)$ denote the maximum of infimums and the minimum of supremums of all braids in the conjugacy class $C(x)$ of x. Given $x = \Delta^u x_1 x_2 \ldots x_k$, in its left canonical form, there are two useful conjugations of x called the *cycling* $c(x)$ and the decycling $d(x)$ defined as follows:

$$\mathbf{c}(x) = \Delta^u x_2 \ldots x_k \tau^u(x_1) = \tau^u(x_1)^{-1} x \tau^u(x_1),$$
$$\mathbf{d}(x) = \Delta^u \tau^u(x_k) x_1 \ldots x_{k-1} = x_k x x_k^{-1}.$$

A braid x is *rigid* if $x_k\tau^u(x_1)$ is left weighted for its weighted form $x = \Delta^u x_1 x_2 \dots x_k$. The *summit set*

$$SS(x) = \{y \in C(x) \mid \inf(y) = \inf_c(x)\}$$

is finite and was introduced by Garside in [11] to solve the conjugacy problem in B_n for the first time. The *super summit set*

$$SSS(x) = \{y \in C(x) \mid \inf(y) = \inf_c(x) \text{ and } \sup(y) = \sup_c(x)\}$$

was introduced by El-Rifai and Morton in [8] to improve Garside's solution. The *reduced super summit set*

$$RSSS(x) = \{y \in C(x) \mid \mathbf{c}^M(y) = y = \mathbf{d}^N(y) \text{ for some positive integers } M, N\}$$

was considered by Lee in his Ph.D. thesis [22] to give a polynomial-time solution to the conjugacy problem in B_4. Finally the *ultra summit set*

$$USS(x) = \{y \in SSS(x) \mid \mathbf{c}^M(y) = y \text{ for some positive integer } M\}$$

was used by Gebhardt in [12] to propose a new algorithm together with experimental data demonstrating the efficiency of his algorithm.

Then we have

$$SS(x) \supset SSS(x) \supset USS(x) \supset RSSS(x).$$

These inclusions are clear from the definitions except the third one which is a consequence of the fact proved by El-Rifai and Morton [8] that if $\inf(x) < \inf_c(x)$ ($\sup(x) > \sup_c(x)$, respectively), then a braid obtained from x by an iteration of cycling (decycling, respectively) has a larger infimum (smaller supremum, respectively).

If x is rigid, then cycling and decycling are predictable and have the same orbit and so we have $RSSS(x) = USS(x)$.

2.2.2 Generating invariant subsets

The convexity theorem by Garside [11], El-Rifai and Morton [8] says that if n-braids x and y are conjugate, there is a sequence of braids $x = x_0, x_1, \dots, x_m = y$ such that for each $1 \le i \le n$, $x_i = a^{-1}x_{i-1}a$ for some $a \in S_n$. The required number of iterations of cycling or decycling to improve an infimum or a supremum is shown to be at most $n(n-1)/2$ by Birman, Ko and Lee [4]. Thus for a given braid x, $\inf_c(x)$ and $\sup_c(x)$ can be obtained within $\ell(x)n(n-1)/2$ iterations of cycling and decycling. Using these facts, the four invariants sets can be generated in finite time since they are finite sets. From an algorithmic point of view, the convexity theorem is not satisfactory since $|S_n| = n!$ is too large.

All the four invariant sets enjoy the property that if $a^{-1}ya \in I$ and $b^{-1}yb \in I$ for $y \in I$ and $a, b \in S_n$ then $(a \wedge b)^{-1}y(a \wedge b) \in I$ where I denotes one of invariant sets. So for $y \in I$, there is a minimal element $a \in S_n$ such that $a^{-1}ya \in I$. Franco and González-Meneses [9] first proved this property for the super summit set and then Gebhardt [12] did it for the ultra summit set. For an invariant set $I(x)$ of a braid x and $y \in I(x)$, this property guarantees the existence of the unique minimal element λ_i such that $\sigma_i \prec \lambda_i \prec \Delta$ and $\lambda_i^{-1}y\lambda_i \in I(x)$. Thus they were able to generate an invariant set more efficiently because there are at most $n-1$ such minimal elements. Unfortunately there is no estimate for the sizes of the invariant sets and so we do not know the complexity of any algorithm based on the generation of an invariant set.

2.3 Hard problems for braid cryptography

Most of cryptographic protocols using the braid groups are based on potential hard mathematical problems as follows.

(1) ***Conjugacy Search Problem (CSP)***
Two braids $a, b \in B_n$ are said to be *conjugate* if there is an element $x \in B_n$ such that $b = x^{-1}ax$. Given two conjugate braids $a, b \in B_n$, the CSP is to find an $x \in B_n$ such that $b = x^{-1}ax$.

(2) ***Multiple Simultaneous Conjugacy Problem (MSCP)***
Given $(a_1, x^{-1}a_1x), \ldots, (a_r, x^{-1}a_rx) \in B_n \times B_n$ for some $x \in B_n$, the MSCP is to find a $y \in B_n$ such that,

$$y^{-1}a_1y = x^{-1}a_1x, \ldots, y^{-1}a_ry = x^{-1}a_rx.$$

(3) ***Diffie-Hellman-Like Conjugacy Problem (DHCP)***
Given a, $x_1^{-1}ax_1$, $x_2^{-1}ax_2$, where x_1 and x_2 belong to some known subgroups of B_n with $x_1x_2 = x_2x_1$, the DHCP is to find $x_2^{-1}x_1^{-1}ax_1x_2$.

All of conjugacy related problems including MSCP and DHCP can easily be reduced to CSP.

2.4 Proposed cryptographic protocols

2.4.1 Commutator key agreement protocol

In 1999, Anshel, Anshel and Goldfeld [1] introduced a key agreement protocol based on combinatorial group theory. They recommended B_n as a promising class of groups for such a construction.

Let S_A and S_B be the following subgroups of B_n:

$$S_A = \langle a_1, a_2, \ldots, a_r \rangle, \quad S_B = \langle b_1, b_2, \ldots, b_s \rangle.$$

Let $a_1, a_2, \ldots, a_r$ and $b_1, b_2, \ldots, b_s \in B_n$ are publicly known.

(1) ***Secret Keys***:
  (a) Alice's Secret Key: $x = W(a_1, a_2, \ldots, a_r) \in S_A$.
  (b) Bob's Secret Key: $y = V(b_1, b_2, \ldots, b_s) \in S_B$.

(2) ***Public Keys***:
  (a) Alice's Public Key: $(x^{-1}b_1x, \ldots, x^{-1}b_sx)$.
  (b) Bob's Public Key: $(y^{-1}a_1y, \ldots, y^{-1}a_ry)$.

(3) ***Shared Key***: $x^{-1}y^{-1}xy$.

Alice and Bob each can compute the shared key because

$$x^{-1}y^{-1}xy = x^{-1}W(y^{-1}a_1y, \ldots, y^{-1}a_ry) = V^{-1}(x^{-1}b_1x, \ldots, x^{-1}b_sx)y.$$

However, the shared keys computed by Alice and Bob may be different as bit strands. Anshel et al. [1], used the *colored Burau representation* for extracting the same bit strand from $x^{-1}y^{-1}xy$.

2.4.2 Diffie-Hellman type key agreement protocol

In 2000 Ko et al. [20] proposed the following key agreement protocol using the DHCP.

Consider the following two subgroups of B_n,

$$LB_n = \langle \sigma_1, \ldots, \sigma_{\lfloor \frac{n}{2} \rfloor - 1} \rangle, \quad \text{and} \quad RB_n = \langle \sigma_{\lfloor \frac{n}{2} \rfloor + 1}, \ldots, \sigma_{n-1} \rangle.$$

Now, for any $x \in LB_n$ and $y \in RB_n$, $xy = yx$. Here $a \in B_n$ is publicly known.

(1) ***Secret Keys***:
   (a) Alice's Secret Key: $x \in LB_n$.
   (b) Bob's Secret Key: $y \in RB_n$.
(2) ***Public Keys***:
   (a) Alice's Public Key: $x^{-1}ax$.
   (b) Bob's Public Key: $y^{-1}ay$.
(3) ***Shared Key***: $y^{-1}x^{-1}axy$.

Since $xy = yx$, Alice and Bob each can compute the shared key. The left-canonical form is used as the final normal form for the shared key.

2.4.3 Braid public-key cryptosystem

In 2000 Ko et al. [20] proposed the following public-key encryption protocol using the DHCP.

Let $h : B_n \to \{0, 1\}^k$ be a secure hash function from the braid group B_n to the message space.

(1) ***Key Generation***:
   (a) Choose a sufficiently complicated braid $u \in B_n$.
   (b) Choose a braid $a \in LB_n$.
   (c) Public key is (u, v) where $v = a^{-1}ua$; Private key is $a \in LB_n$.
(2) ***Encryption***: Given a message $m \in \{0, 1\}^k$ and a public key (u, v),
   (a) Choose a braid $b \in RB_n$;
   (b) The ciphertext is (c, d) where $c = b^{-1}ub$ and $d = h(b^{-1}vb) \bigoplus m$.
(3) ***Decryption:*** Given the ciphertext (c, d) and private key a, obtain $m = h(a^{-1}ca) \bigoplus d$.

Since $ab = ba$ for $i = 1, 2$, $h(a^{-1}ca) \bigoplus d = h(b^{-1}vb) \bigoplus h(b^{-1}vb) \bigoplus m = m$ and the decryption recovers the original message m.

2.5 Attacks on the conjugacy and related problems

We now review the history of major attacks on the conjugacy and related problems since braid cryptosystems were proposed.

2.5.1 Using invariant sets

The cardinality of any of conjugacy invariant sets is unknown and hence the complexity of any attack based on an invariant set is not known either. Nevertheless this kind of attack would be much more effective for MSCP than for the ordinary CSP.

Lee and Lee [23] applied super summit set in solving the MSCP. Given $(a_1, \ldots, a_r)$ and $(c_1, \ldots, c_r)$ where $c_i = x^{-1}a_i x$ and $\inf(x) = 0$ or 1, their method is to transform $(c_1, \ldots, c_r)$ into $(d_1, \ldots, d_r)$, where all d_i's are conjugates of a_i's by a single conjugator whose word-length is smaller than $|x|$ and then to find an $x' \in B_n$ satisfying $d_i = (x')^{-1}a_i x'$, $\forall i$. This attack can be improved as much as done in [9].

2.5.2 Length-based techniques

Given an element x and a conjugate $a^{-1}xa$, it is sometimes possible to recover the conjugating element a by repeatedly conjugating $a^{-1}xa$ with short braids t such that $t^{-1}(a^{-1}xa)t$ is shorter or less complicated than $a^{-1}xa$. They are frequently effective if x and a are chosen randomly. However, attacks of this kind can be prevented by a careful choice of x and a; the idea is that x and $a^{-1}xa$ should have the same length or complexity.

In the commutator key agreement protocol, the public key of Alice is $(x^{-1}b_1x, \ldots, x^{-1}b_sx)$ where $x = W(a_1, a_2, \ldots, a_r)$ is Alice's secret key. Hughes and Tannenbaum [16] and Garber, Kaplan, Teicher, Tsaban and Vishne [10] proposed some length based attacks on the commutator key agreement protocol for the case in which $|a_i|$ for all a_i was large. The natural method to recover x from the public key is to find $a \in \{a_1, \ldots, a_r, a_1^{-1}, \ldots a_r^{-1}\}$ such that $|ax^{-1}b_jxa^{-1}|$ is much smaller than $|x^{-1}b_jx|$ for the majority of $j \in \{1, \ldots, s\}$. By repeating this, x is recovered.

2.5.3 Using representations of braid groups

Let R be a ring and $GL_k(R)$ be the general linear group. A *representation* of a group is a homomorphism from the group to $GL_k(R)$. A representation is said to be *faithful*, if it is injective. The Burau and the Lawrence Krammer representations of the braid groups have been used to study the conjugacy and related problems.

The Burau representations, $\rho_n : B_n \to GL_n(\mathbb{Z}[t^{\pm 1}])$ is defined by

$$\rho_n(\sigma_i) = \begin{bmatrix} \mathbf{I}_{i-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \begin{bmatrix} 1-t & t \\ 1 & 0 \end{bmatrix} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_{n-i-1} \end{bmatrix}$$

This representation is not faithful for $n \geq 5$. For $\alpha \in B_n$, $\rho_n(\alpha)$ is called the *Burau matrix*.

Hughes [15] proposed a heuristic method using ρ_n to solve MSCP. Let $\{(a_1, c_1), \ldots, (a_r, c_r) : c_i = x^{-1}a_ix, \text{for } 1 \leq i \leq r\} \subset B_n \times B_n$ be known, where $x \in B_n$ is unknown. His method is to compute $\rho_B(x)$ from $(a_1, c_1), \ldots, (a_r, c_r)$ and then to compute $x \in B_n$ from $\rho_B(x)$. He used some very fast linear algebra solvers called banded solvers to compute $\rho_B(x)$. Hughes empirically showed that, the first column containing the highest degree entry in $\rho_B(x)$ tends to indicate a last Artin generator of x, especially when $|x|$ is small. Using this observation he computed x from $\rho_B(x)$ generator by generator.

Lee and Park also proposed solutions to CSP and DHCP using Burau representation by proposing two improvements to Hughes algorithm [24]. They showed that the private-key could be recovered from the public-key for several parameters with significant probability in a reasonable time. Comparing with his algorithm one is more efficient with same success rate and the other has higher success rate with less efficiency. They, also theoretically proved the Hughes empirical result that, first column containing the highest degree entry in $\rho_B(x)$ tends to indicate a last Artin generator of x, especially when $|x|$ is small and they computed x from $\rho_B(x)$ generator by generator like Hughes.

The Lawrence-Krammer representation $\rho_K : B_n \to GL_{\frac{n(n-1)}{2}}(\mathbb{Z}[t^{\pm 1}, q^{\pm 1}]) = Aut(V_0)$, where V_0 is the free module of rank $\frac{n(n-1)}{2}$ over $\mathbb{Z}[t^{\pm 1}, q^{\pm 1}]$. With respect to the free basis $\{x_{ij}\}_{1 \leq i < j \leq n}$ of V_0, the image of each Artin generator under ρ_K is given by

$$\rho_K(\sigma_k)(x_{ij}) = \begin{cases} tq^2 x_{k,k+1}, & \text{if } i = k, j = k+1; \\ (1-q)x_{i,k} + qx_{i,k+1}, & \text{if } j = k, i < k; \\ x_{ik} + tq^{k-i+1}(q-1)x_{k,k+1}, & \text{if } j = k+1, i < k; \\ tq(q-1)x_{k,k+1} + qx_{k+1,j}, & \text{if } i = k, k+1 < j; \\ x_{kj} + (1-q)x_{k+1,j}, & \text{if } i = k+1, k+1 < j; \\ x_{i,j}, & \text{if } i < j < k \text{ or } k+1 < i < j; \\ x_{ij} + tq^{k-i}(q-1)^2 x_{k,k+1}, & \text{if } i < k < k+1 < j. \end{cases}$$

where t and q are variables. This representation is faithful for all n. For $\alpha \in B_n$, $\rho_K(\alpha)$ is called the *Lawrence-Krammer matrix*.

Cheon and Jun [6] used the Lawrence-Krammer representation ρ_K to recover the shared key in DHCP. The problem is as follows: given $a, x^{-1}ax, y^{-1}ay \in B_n$, where $x \in LB_n$ and $y \in RB_n$, find $x^{-1}y^{-1}ayx$. Their method is to compute $\rho_K(x^{-1}y^{-1}ayx)$ and then to recover $x^{-1}y^{-1}ayx$ from it.

They first determine $\rho_K(x)$ from $(a, x^{-1}ax)$ by solving $X\rho_K(b) = \rho_K(a)X$, where $b = x^{-1}ax$, $X = \rho_K(x)$ and using the constraint,

$$X\rho_K(\sigma_i) = \rho_K(\sigma_i)X, \quad \forall i > \lfloor \frac{n}{2} \rfloor.$$

Now $\rho_K(x^{-1}y^{-1}ayx)$ is determined using the relation,

$$\rho_K(x^{-1}y^{-1}ayx) = X^{-1}\rho_K(y^{-1}ay)X.$$

Let $z = x^{-1}y^{-1}ayx$. Then z is recovered from $\rho_K(z)$ by using the Laurent series of ρ_K with respect to t, namely

$$\rho_K(z) = \sum_{i=u}^{u+l} A_i t^i, \quad \text{where } A_i \in M_{\frac{n(n-1)}{2} \times \frac{n(n-1)}{2}}(\mathbb{Z}[q^{\pm 1}]), \quad A_u, A_{u+l} \neq \mathbf{O}_{\frac{n(n-1)}{2}}.$$

The overall time complexity of the scheme is $O(n^{14.4}|x|^{3.2})$. We note that this attack has not only relatively high complexity but also is hard to be implemented because one has to deal with large matrices with entries in a noncommutative ring.

2.5.4 Attacks based on removing redundancy

We now review some other attacks that are effective when keys are randomly generated.

A simple heuristic approach to the conjugacy problem in braid groups was described by D. Hofheinz and R. Steinwandt [14]. Although it did not provide a general solution to the conjugacy problem, it demonstrated that various proposed key parameters for braid cryptographic primitives do not offer acceptable cryptographic security.

A. Myasnikov, V. Shpilrain and A. Ushakov proposed a practical heuristic solution for DHCP in Crypto 2005 [26]. Using the attack they were able to break the key exchange protocol of Ko et al. in about 150 min with over 95% success rate for typical parameters. One of the ideas behind the attack was using Dehornoy's handle reduction method as a counter measure to diffusion provided by the Garside normal form and as a tool for simplifying braid words. Also, they solved the decomposition problem in braid groups rather than the conjugacy problem.

S. Maffre [25] proposed an algorithm for the conjugacy problem which gives a partial factorization of the secret: a divisor and a multiple. The efficiency of his attack depends on

the random generator used to create the key. His idea is related to the one of Hofheinz and Stenwandt [14]. They proposed an algorithm which computes a prefix of the secret ($d \prec a$): afterwards, if the canonical length of $d^{-1}a$ is 1, they map the problem to the symmetric group and conclude. But Maffre proposed an algorithm which computes the multiple of a secret: knowing (x, axa^{-1}) and the canonical length of a, it computes m such that a is a prefix of m ($a \prec m$) and $axa^{-1} = m\tilde{x}m^{-1}$. This algorithm exploits the decomposition of braids into products of canonical factors. It allows also to obtain a prefix of the secret: $d \prec a$. The knowledge of the canonical length of the secret is required.

All of these attacks take advantage of redundancy that every randomly generated instances of CSP or DHCP must have. In the next section, we will show that a braid chosen randomly has extremely simple USS and RSSS in overwhelming probability. A random instance for DHCP has even more redundancy due to half braiding.

3 Analysis of the current attacks on braid cryptosystem

Except attacks based on a finite invariant conjugacy set or a representation of the braid groups, all other attacks were based on heuristic algorithms removing redundancies and therefore the attacks were concentrated on instances that are randomly generated by some algorithms. In this section, we introduce two natural ways of randomly generating braids and compare them. Then we discuss why these heuristic attacks were effective and easy by presenting a fast solution to the conjugacy problem that should be successful for random instances in overwhelming probability.

3.1 Generating random braids

In the braid groups given by Artin's presentation, there are two natural ways of generating random words as follows:

(1) Choosing a sequence of random Artin generators and concatenate them;
(2) Choosing a sequence of random permutations and multiply them.

The fastest solution to the word problem in the braid group is to write a braid word in its weighted form. In order to have a unique representation of braid words, braids are coded in their unique weighted form. Factors in a weighted form can be described by either a permutation or a product of Artin generators. Since the average number of inversions in an n-permutation is $\frac{n(n-1)}{4}$, so is the average word length of a factor in a braid generated by multiplying random permutations. If a factor is represented a product of Artin generator, we need on average $\frac{n(n-1)}{4}$ of $\log n$-bit integers. If a factor is represented by a permutation, we need an n-tuple of $\log_n$-bit integers. Thus when we digitize a random braid from permutations, it is more efficient to use permutations than Artin generators, especially as n becomes large. On the contrary, we will see that it is more efficient to use Artin generators to digitize a random braid from Artin generators.

3.2 Random braids from permutations

Most of the attacks on braid cryptosystem have been carried out on braids generated by picking random permutation braids and multiplying them. A random permutation is generated by for $i = 1, 2, \ldots, n-1$, picking a random number between i and n. Then each permutation can be chosen in the probability $\frac{1}{n!}$. Recently, Ko and Lee [18] showed that a random braid from

permutations is already rigid and its USS (=RSSS) contains at most 2 orbits in overwhelming probability. Using this fact, they were able to propose a fast algorithm for the conjugacy problem that is successful on randomly generated instances in overwhelming probability.

For integers $1 \leq i < j \leq n$, a positive n-braid x is said to begin with an inversion (i, j) if its head $H(x)$ exchanges i and j as a permutation.

For permutation n-braids $x_1, x_2, \ldots, x_k$ chosen randomly, let $d(n, k)$ denote the average contribution by the last factor x_k, to the set of Artin generators that the product $x_1 x_2 \ldots x_k$ begins with. They found an expression in terms of n and k of an upper bound for $d(n, k)$ and then showed that $d(n, k)$ converges to 0 either as k increases for fixed n or as n increases for a fixed $k \geq 3$. Also they showed that the larger the n, the faster it converges to 0 as k increases. Then they proved the following:

Theorem 3.1 [18] *For $k \geq 5$ and randomly chosen permutation n-braids $x_1, \ldots, x_k$ and a random integer u, let $x = \Delta^u x_1 \cdots x_k$. Then*

(1) *The probability that $\mathbf{c}^j(x)$ is rigid for some $j \leq \lfloor \frac{k}{2} \rfloor$ is greater than $1 - 2d(n, \lfloor \frac{k}{4} \rfloor)$;*
(2) *The probability that the number of orbits in USS(x)=RSSS(x) is at most two is also greater than $1 - 2d(n, \lfloor \frac{k}{4} \rfloor)$*

Given a random braid $x \in B_n$, an algorithm to generate $USS(x)$ proceeds as follows:

(1) Compute $y = \mathbf{c}^j(x)$ where $j = \lfloor \frac{\ell(x)}{2} \rfloor$.
(2) Output either $\{\mathbf{c}^i(y),\ \tau(\mathbf{c}^i(y)) \mid 0 \leq i \leq \ell(x) - 1\}$ if $\inf(y)$ is even, or $\{\mathbf{c}^i(y) \mid 0 \leq i \leq 2\ell(x) - 1\}$ if $\inf(y)$ is odd.

Since the operation to convert into a weighted form has running time $\mathcal{O}(k^2 n \log n)$ and this algorithm requires k cycling operations, the overall running time is $\mathcal{O}(k^3 n \log n)$. For a given n-braid x of k random permutations, it generates $USS(x)$ successfully with probability greater than $1 - 2d(n, \lfloor \frac{k}{4} \rfloor)$. For random braids from permutations, the conjugacy problem generically becomes easier as either the braid index or the canonical length increases, contrary to the common belief.

3.3 Pseudo-Anosov Braids

As a homeomorphism of a 2-dimensional disk that preserves n distinct interior points and fixes the boundary of the disk, an n-braid x is isotopic to one of the following three dynamic types known as the Nielson-Thurston classification:

(1) *periodic*, if x^p is the identity for some non-negative integer p;
(2) *reducible*, if x preserves a set of disjointly embedded circles;
(3) *pseudo-Anosov*, if it is neither periodic nor reducible.

The dynamic types are invariant under conjugation and taking powers. Ko and Lee [19] proposed a polynomial-time algorithm to decide the dynamic type of a given braid. The algorithm take advantage of the following facts:

Theorem 3.2 (1) [3,19] *A pseudo-Anosov braid becomes rigid up to iteration of cycling and decycling and taking powers.*
(2) [19] *A reducible, rigid braid has a standard reduction circles up to a conjugation by a permutation braid.*

Circles preserved by a reducible braid are *standard* if they really look like round circles and can be detected by a polynomial-time algorithm. The conjugacy problem for periodic

braids is trivial. To solve the conjugacy problem for reducible braids, an approach via finite conjugacy invariant sets may be difficult because USSS or RSSS can be huge. Nonetheless we think that the problem becomes easier after reduction circles are recognized. Pseudo-Anosov braids are generic among three dynamical types and behave similar to random braids in high probability. But there are some difficult instances of the conjugacy problem involving pseudo-Anosov braids as we will see in Sects. 4.3 and 4.4.

3.4 Random Braids from Artin generators

We first consider the density of braid words generated by concatenating Artin generators chosen randomly. The following tables shows the average word length of an n-braid of k canonical factors in its weighted form of a random braid from Artin generators. The experiments were carried out 10^5 times. The second column is the average word length of a factor obtained from random permutation. The column with the header $\frac{w(k)}{k}$ shows the average word length of a factor in random n-braids of the supremum k that are generated by concatenating random Artin generators. For each braid index n, $\frac{n(n-1)}{4}$ is much larger than $\frac{w(k)}{k}$ and this means random braids from permutation is much denser than those from Artin generators. This reconfirms Dehornoy's observation in [7] that when we generate random braid words, the average canonical length of braids from random Artin generators are much larger than those from random permutations. Furthermore $\frac{w(k)}{k}$ is even smaller than n and this implies that it is more efficient to represent each factor by a word of Artin generators than by a permutation.

n	$\frac{n(n-1)}{4}$	$\frac{w(1)}{1}$	$\frac{w(2)}{2}$	$\frac{w(3)}{3}$	$\frac{w(4)}{4}$	$\frac{w(5)}{5}$	$\frac{w(6)}{6}$	$\frac{w(7)}{7}$	$\frac{w(8)}{8}$	$\frac{w(9)}{9}$	$\frac{w(10)}{10}$
5	5	2.71	2.92	3.03	3.11	3.16	3.20	3.22	3.24	3.26	3.27
10	22.50	4.08	4.77	5.18	5.45	5.64	5.78	5.89	5.98	6.04	6.09
20	95	5.83	7.43	8.48	9.21	9.75	10.15	10.47	10.72	10.93	11.10
50	612.50	9.16	13.16	16.05	18.24	19.89	21.17	22.20	23.06	23.77	24.37

Let us now look at the ultra summit and the reduced super summit sets of random braids from Artin generators. Our experiment on the number of cycling orbits in the USS and RSSS of random braids from Artin generators are tabulated below. The experiments were done by generating a random n-braids of the supremum k by concatenate random Artin generators as long as the supremum of its weighted form is k. The average values of the infimum (inf), maximum of infimums of all braids in the conjugacy class ($\inf_c$), supremum (sup), the minimum of supremums of all braids in the conjugacy class ($\sup_c$), the number of orbits in the ultra summit set and the number of orbits in the reduced super summit set are tabulated. The experiments were carried out 1,000 times.

n	$k = \sup$	inf	$\inf_c$	$\sup_c$	$\overline{USS}$	$\overline{RSSS}$
5	30	0.668	0.783	29.154	1.593	1.593
10	20	0	0	18.798	2.068	2.068
15	20	0	0	18.708	2.430	2.430
20	15	0	0	13.777	4.211	4.211
25	15	0	0	13.785	7.011	7.011
30	15	0	0	13.801	11.411	11.411

In the table we can observed that the numbers of orbits in USS and RSSS, denoted by $\overline{USS}$ and $\overline{RSSS}$ repetively, are the same. This means that these random braids become rigid after iteration of cycling and decyling in overwhelming probability. The number of orbits in USS and RSSS are small but are not 2 as for random braids from permutations. This means it is

easier to obtain a braid with large RSSS using random braids from Artin generators. In the previous section we saw that it is almost impossible to obtain a braid with large RSSS from random braids from permutations. In fact we will give an example of pseudo-Anosov braids generated by concatenating random Artin generators that has very large RSSS.

4 Potential hard instances of conjugacy problem

The generation of the finite conjugacy invariant sets USS or $RSSS$ is state-of-the-art when we try to solve the conjugacy problem for all possible instances. Since $RSSS$ is the smallest invariant set, we need to consider $RSSS$ more seriously in the following discussion. Thus braids with large $RSSS$ are obvious candidates for hard instances of the conjugacy problem. In order to make applications to cryptography, we need security parameters that control the difficulty of the conjugacy problem. The obvious choices are the braid index n and the canonical length k. For random braids discussed in the previous section, the distribution of braids with large $RSSS$ is getting sparse as either n or k increase and so random instances become easier to solve.

USS or $RSSS$ are disjoint unions of orbits. Since each orbit in USS or $RSSS$ is not complex and is rather easy to generate, the difficulty of using USS or $RSSS$ to solve the conjugacy problem depends more on the number of orbits than on the size of each orbit. When we suggest a class of braids that give potentially hard instances for the conjugacy problem, we need to demonstrate that the number of orbits in USS or in $RSSS$ increases rapidly as either n or k increase so that n and k play the role of security parameter.

We now construct three distinct classes of braids for which the numbers of orbits in USS and $RSSS$, denoted by $\overline{USS}$ and $\overline{RSSS}$, are large and rapidly increasing with respect to the braid index n and the canonical length k.

4.1 Split Braids

In this section, we consider a class of reducible braids called *split braids*. For $i \in \{1, 2, \ldots, n-1\}$, let B_n^i be the subgroup of B_n generated by the elements $\sigma_1, \ldots, \sigma_{i-1}, \sigma_{i+1}, \ldots, \sigma_{n-1}$. The elements of B_n^i are called *split braids of two components*. We give some examples of such braids for various values of the braid index n and canonical length k.

In the following examples, braids are given in their weighted forms where each factor is a permutation over $\{1, 2, \ldots, n\}$. All of the examples are made rigid so that they are already in their $RSSS$.

Example 1 The braid in B_8^4 given by

$$(4\,3\,2\,1\,7\,8\,6\,5)\ (3\,4\,2\,1\,7\,5\,6\,8)\ (2\,3\,1\,4\,5\,8\,5\,7)$$
$$(1\,2\,3\,4\,8\,6\,7\,5)\ (1\,2\,3\,4\,8\,5\,7\,6)$$

has $\inf = \inf_c = 0$, $\sup = \sup_c = 5$, $\overline{USS} = 136$ and $\overline{RSSS} = 104$.

Example 2 The braid in B_8^4 given by

$$(4\,3\,2\,1\,6\,8\,5\,7)\ (4\,3\,2\,1\,8\,6\,7\,5)\ (4\,3\,2\,1\,8\,5\,7\,6)\ (4\,2\,3\,1\,7\,8\,6\,5)$$
$$(2\,1\,3\,4\,8\,5\,6\,7)\ (1\,2\,3\,4\,5\,7\,8\,6)\ (1\,2\,3\,4\,6\,8\,5\,7)\ (1\,2\,3\,4\,5\,7\,8\,6)$$

has $\inf = \inf_c = 0$, $\sup = \sup_c = 8$, $\overline{USS} = 5890$ and $\overline{RSSS} = 3072$.

Example 3 The braid in B_{10}^5 given by

$$(5\,4\,3\,2\,1\,10\,6\,8\,7\,9)\ (5\,4\,3\,2\,1\,8\,10\,6\,7\,9)\ (4\,3\,5\,2\,1\,7\,10\,6\,9\,8)$$
$$(1\,2\,3\,4\,5\,10\,8\,9\,7\,6)\ (1\,2\,3\,4\,5\,9\,8\,7\,10\,6)$$

has $\inf = \inf_c = 0$, $\sup = \sup_c = 5$, $\overline{USS} = 5028$ and $\overline{RSSS} = 1160$.

Example 4 The braid in B_{10}^5 given by

$$(5\,4\,3\,2\,1\,6\,8\,10\,7\,9)\ (5\,4\,3\,2\,1\,6\,9\,8\,10\,7)\ (1\,4\,3\,2\,5\,8\,10\,6\,7\,9)$$
$$(1\,3\,2\,4\,5\,9\,10\,6\,8\,7)\ (3\,5\,1\,2\,4\,6\,10\,8\,7\,9)\ (4\,5\,2\,3\,1\,8\,10\,7\,9\,6)$$
$$(1\,2\,3\,4\,5\,10\,7\,6\,9\,8)\ (1\,2\,3\,4\,5\,9\,8\,10\,6\,7)$$

has $\inf = \inf_c = 0$, $\sup = \sup_c = 5$, $\overline{USS} = 78105$ and $\overline{RSSS} = 27790$.

We remark that examples above has the following characteristics:

(1) One of split components has the infimum 1 and the other has the infimum 0;
(2) Canonical lengths of split components are relatively prime.

We can construct reducible braids from split braids by multiplying a pure outer braid while maintaining $\overline{RSSS}$ and $\overline{RSSS}$.

4.2 Cabled braids

Let α be an n-braid such that some $1 \leq i \leq n$, the strand of α that starts at the i-th spot, also ends at the i-th spot. Let $\tilde{\alpha}$ be the $(n+1)$-braid, obtained from α by adding a strand on the right of the i-th strand and parallel to it. The crossings made by the i-th and the new strands with the other strands are identical. We call such a braid a *cabled* braid obtained from α along the i-th strand.

In the following examples, the braids are given in their weighted forms.

Example 1

$$\alpha = (5\,4\,3\,2\,1)\ (5\,4\,2\,3\,1)\ (3\,2\,5\,4\,1)\ (5\,2\,1\,4\,3)\ (5\,1\,2\,3\,4)\ (1\,2\,3\,5\,4)$$

and

$$\tilde{\alpha} = (6\,4\,5\,3\,2\,1)\ (6\,5\,2\,3\,4\,1)\ (3\,2\,5\,6\,4\,1)\ (6\,2\,1\,5\,3\,4)\ (6\,1\,2\,3\,4\,5)\ (1\,2\,3\,4\,6\,5).$$

Example 2

$$\alpha = (5\,4\,3\,2\,1)\ (3\,2\,1\,5\,4)\ (2\,5\,1\,4\,3)\ (4\,3\,5\,2\,1)\ (5\,4\,2\,1\,3)\ (5\,3\,4\,2\,1)\ (2\,1\,4\,5\,3)$$
$$(1\,2\,5\,3\,4)\ (1\,3\,4\,5\,2)\ (1\,3\,2\,4\,5)\ (1\,3\,2\,4\,5).$$

and

$$\tilde{\alpha} = (5\,6\,4\,3\,2\,1)\ (3\,2\,1\,6\,4\,5)\ (2\,6\,1\,4\,5\,3)\ (5\,4\,6\,2\,3\,1)\ (6\,4\,5\,2\,1\,3)\ (6\,4\,5\,2\,3\,1)$$
$$(3\,1\,2\,5\,6\,4)\ (1\,2\,3\,6\,4\,5)\ (1\,2\,4\,5\,6\,3)\ (1\,2\,4\,3\,5\,6)\ (1\,2\,4\,3\,5\,6)$$

Example 3

$$\alpha = (9\,8\,7\,6\,5\,4\,3\,2\,1)\ (9\,8\,7\,4\,3\,2\,6\,1\,5)\ (4\,9\,7\,5\,6\,3\,2\,8\,1)\ (3\,8\,9\,1\,2\,4\,5\,6\,7)$$

and

$$\tilde{\alpha} = (10\,9\,8\,7\,5\,6\,4\,3\,2\,1)\ (10\,9\,8\,5\,3\,4\,2\,7\,1\,6\,)\ (4\,10\,7\,8\,5\,6\,3\,2\,9\,1)$$
$$(3\,9\,10\,1\,2\,4\,5\,6\,7\,8).$$

Example 4

$$\alpha = (9\,8\,7\,6\,5\,4\,3\,2\,1)\ (9\,5\,1\,4\,3\,2\,8\,7\,6)\ (7\,8\,5\,6\,4\,9\,2\,3\,1)\ (1\,8\,9\,2\,5\,6\,3\,4\,7)$$
$$(1\,3\,5\,6\,4\,7\,8\,2\,9)\ (1\,8\,2\,5\,3\,4\,6\,7\,9)\ (1\,3\,5\,7\,4\,6\,9\,2\,8)\ (7\,9\,4\,6\,3\,5\,2\,8\,1)$$
$$(2\,6\,9\,5\,8\,7\,1\,4\,3)\ (3\,2\,9\,1\,6\,5\,8\,4\,7).$$

and

$$\tilde{\alpha} = (10\,9\,8\,7\,6\,5\,3\,4\,2\,1)\ (10\,6\,1\,2\,5\,4\,3\,9\,8\,7)\ (7\,8\,9\,5\,6\,4\,10\,2\,3\,1)$$
$$(1\,9\,10\,2\,6\,7\,3\,4\,5\,8)\ (1\,3\,5\,6\,7\,4\,8\,9\,2\,10)\ (1\,9\,2\,6\,3\,4\,5\,7\,8\,10\,)$$
$$(1\,3\,5\,6\,8\,4\,7\,10\,2\,9)\ (8\,10\,5\,7\,3\,4\,6\,2\,9\,1)\ (2\,6\,9\,10\,5\,8\,7\,1\,4\,3)$$
$$(3\,2\,10\,1\,6\,5\,9\,4\,7\,8).$$

These examples have the properties as in the following table:

Example	n	k	$\overline{USS}(\alpha)$	$\overline{RSSS}(\alpha)$	$\overline{USS}(\tilde{\alpha})$	$\overline{RSSS}(\tilde{\alpha})$
1	5	5	3	3	2983	9
2	5	10	1	1	12609	18
3	9	3	2	2	36456	3
4	10	10	18	18	> 611000	35

We remark that the cycling orbit of a cabled braid $\tilde{\alpha}$ of α has length 2 and $\overline{USS}(\tilde{\alpha}) \geq |SSS(\alpha)|/2$. Thus $\overline{USS}(\tilde{\alpha})$ can be huge. However $\overline{RSSS}(\tilde{\alpha})$ is rather small and so this class will not pose hard instances for the conjugacy problem.

4.3 Quasi-reducible braids

In this section, we consider a class of pseudo-Anosov braids that are almost reducible so that they still possess some of characteristics of reducible braids, such as large $\overline{USS}$ and $\overline{RSSS}$. Thus we call braids in this class *quasi-reducible*. It seems natural to believe that pseudo-Anosov braids give rise to harder instances than reducible braids do when their invariant sets are similar in size. This is because the braid index that is one of security parameters can be reduced if some of reducing circles of a reducible braid are revealed. Thus the class of quasi-reducible braids is interesting.

It is difficult to give a precise definition of quasi-reducible braids other than they are pseudo-Anosov braids with some characteristics of reducible braids. Basically a pseudo-Anosov braid is quasi-reducible if it becomes reducible after nullifying "few" crossings. The word "few" may depend on how many inner braids are formed. In the following examples, pseudo-Anosov braids split into two inner braids after nullifying crossings in one factor.

Example 1 The 8-braid given by

$$(4\,3\,1\,2\,6\,8\,5\,7)\ (3\,8\,1\,2\,5\,4\,7\,6)\ (1\,3\,2\,8\,4\,6\,5\,7)\ (3\,4\,2\,7\,8\,1\,6\,5)$$

has $k = 4$, $\overline{RSSS} = 60$, and $|RSSS| = 240$.

Example 2 The 8-braid given by

$$(4\,2\,3\,1\,8\,6\,7\,5)\ (1\,3\,4\,2\,8\,5\,7\,6)\ (2\,4\,1\,3\,6\,8\,5\,7)\ (2\,3\,4\,1\,8\,6\,7\,5)$$
$$(4\,1\,2\,3\,7\,6\,8\,5)\ (3\,4\,5\,2\,8\,7\,1\,6)\ (2\,4\,1\,3\,5\,8\,7\,6)\ (2\,4\,8\,1\,3\,7\,6\,5).$$

has $k = 8$, $\overline{RSSS} = 72$, and $|RSSS| = 576$.

Example 3 The 10-braid given by

(5 4 2 3 1 9 6 8 7 10) (1 3 5 4 2 6 10 8 7 9) (3 6 4 5 2 9 10 1 8 7)
(5 3 2 4 10 1 7 9 6 8)

has $k = 4$, $\overline{RSSS} = 130$, and $|RSSS| = 520$.

Example 4 The 10-braid given by

(5 4 3 1 2 10 9 7 8 6) (3 4 2 5 1 10 6 8 7 9) (5 2 1 3 4 8 10 7 9 6)
(5 2 3 4 1 6 10 8 9 7) (4 1 3 10 2 6 8 7 9 5) (2 4 3 1 6 5 9 7 8 10)
(5 3 4 2 7 1 6 9 8 10) (1 4 3 5 2 8 7 10 6 9)

has $k = 8$, $\overline{RSSS} = 840$, and $|RSSS| = 6720$.

Let x be one of examples above. Then we note that x has the following characteristics:

(1) x is a pseudo-Anosov, rigid braid;
(2) x becomes a split braid by nullify one crossing;
(3) There are positive braids $\gamma,\ \gamma'$ such that
  (i) $\gamma x \gamma^{-1}$ and $\gamma' x \gamma'^{-1}$ are in $RSSS(x)$;
  (ii) $\gamma\gamma' = \gamma'\gamma$;
  (iii) $\ell(\gamma),\ \ell(\gamma') \geq 2$.

4.4 Quisi-reducible braid from artin-generator

The following example were found during an experiment to generate random braids by concatenating Artin generators. Since this method generates random braids that are sparser than the method using permutations, we have a fair amount of chance to obtain a quasi-reducible braid with large $\overline{RSSS}$.

Example 1 The 10-braid given by

(1 3 2 5 4 7 6 9 8 10) (1 3 2 5 4 9 7 8 6 10) (2 4 3 6 1 10 5 8 7 9)
(1 3 10 2 5 4 7 6 9 8)

has the following properties:

(1) It is rigid and $k = 4$;
(2) $\overline{RSSS} = 780$;
(3) $\beta\sigma_5^{-1}$ is split where $\beta = (\sigma_4^2\sigma_6\sigma_7\sigma_8\sigma_9)\alpha(\sigma_4^2\sigma_6\sigma_7\sigma_8\sigma_9)^{-1}$.

Example 2 The 10-braid given by

(2 4 1 3 5 7 6 8 10 9) (5 1 3 2 4 7 6 8 10 9) (1 3 2 5 4 7 6 8 10 9)
(1 3 2 6 5 8 4 7 9 10) (1 3 2 4 10 5 7 6 8 9)

has the following properties:

(1) It is rigid and $k = 5$;
(2) $\overline{RSSS} = 1008$;
(3) $\beta\sigma_5^{-1}$ is split where $\beta = (\sigma_4\sigma_6\sigma_7\sigma_8\sigma_9)\alpha(\sigma_4\sigma_6\sigma_7\sigma_8\sigma_9)^{-1}$.

5 Conclusion

In this article, we first discussed various attacks on the conjugacy problems of the braid groups and the cryptosystems based on them. We explained why some of attacks were successful by showing that the conjugacy problem of the braid groups is rather easy for instances generated randomly. We proposed several ways of generating instances for the conjugacy problem that seems secure under the most sophisticated attack based on current knowledge. These classes of braids should be interesting not only to cryptoanalysts but also to mathematicians who work on the braid groups.

References

1. Anshel I., Anshel M., Goldfeld D.: An algebraic method for public-key cryptography. Math. Res. Lett. **6**, 287–291 (1999).
2. Birman J.S.: Braids, links and mapping class groups, Annals of Math. Study 82, Princeton University Press (1974).
3. Birman J.S., Gebhardt V., González-Mensese J.: Conjugacy in Garside groups I: Cyclings, Powers, and Rigidity, arXiv:math.GT/0605230
4. Birman J.S., Ko K.H., Lee S.J.: The infimum, supremum, and geodesic length of a braid conjugcy class. Adv. Math. **164**(1), 41–56 (2001).
5. Cha J.C., Ko K.H., Lee S.J., Han J.W., Cheon J.H.: An efficient implementation of braid groups. Advances in Cryptology: Proceedings of ASIACRYPT 2001, Lecture Notes in Computer Science, Springer-Verlag, vol 2248 pp. 144–156 Springer-verlag (2001).
6. Cheon J.H., Jun B.: A polynomial time algorithm for the braid Diffie-Hellman conjugacy problem. Advances in Cryptology: Proceedings of CRYPTO 2003, Lecture Notes in Computer Science, vol 2729, pp. 212–225. Springer-Verlag-2003.
7. Dehornoy P.: Braid-based cryptography. Contemp. Math. **360**, 5–33 (2004).
8. El-Rifai E.A, Morton H.R.: Algorithms for positive braids. Quart. J. Math. Oxford Ser. **45**(2), 479–497 (1994).
9. Franco N., Gonazález-Meneses J.: Conjugacy problem for braid groups and Garside groups. J. Algebra **266**(1), 112–132 (2003).
10. Garber D., Kaplan S., Teicher M., Tsaban B., Vishne U.: Length-based conjugacy search in the Braid groups. http://arXiv.org/abs/math.GR/0209267
11. Garside F.A.: The braid group and other groups. Quart. J. Math. Oxford Ser. **20**(2), 235–254 (1969).
12. Gebhardt V.: A new approach to the conjugacy problem in Garside groups, to appear in J. Algebra, (2005).
13. Gebhardt V.: Conjugacy search in braid groups, from a braid-based cryptography point of view, applicable algebra in engineering. Commun Computi. **17**(3–4), 219–238 (2006).
14. Hofheinz D., Steinwandt R.: A practical attack on some braid group based cryptographic primitives, 6th International Workshop on Practice and Theory in Public Key Cryptography: Proceedings of PKC 2003, Lecture Notes in Computer Science, vol 2567, pp. 187–198. Springer Verlag (2002).
15. Hughes J.: A linear algebraic attack on the AAFG1 braid group cryptosystem, ACISP 2002, Lecture Notes in Computer Science, vol 2384, pp. 176–189. Springer-Verlag (2002).
16. Hughes J., Tannenbaum A.: Length-based attacks for certain group based encryption rewriting systems, Workshop SECI02, *Sécurité* de la Communication sur Internet, Sept. 2002.
17. Ko K.H., Choi D.H., Cho M.S., Lee J.W.: New signature scheme using conjugacy problem, Available at: http://eprint.iacr.org/2002/168.pdf
18. Ko K.H., Lee J.W.: A fast algorithm to the conjugacy problem on generic braids, to appear in the proceedings of Knot theory for Scientific Objects, March, 2006, Osaka, Japan.
19. Ko K.H., Lee J.W.: A polynomial-time solution to the reducibility problem, arXiv:math.GT/0610746.
20. Ko K.H., Lee S.J., Cheon J.H., Han J.W., Kang J.S., Park C.S.: New public-key cryptosystem using braid groups. Advances in Cryptology: Proceedings of CRYPTO 2000. Lecture Notes in Computer Science, vol 1880, pp. 166–183. Springer-Verlag (2000).
21. Lee E.: Braid groups in cryptology. IEICE Trans. Fundamentals, E **87**A(5), 986–992 (2004).
22. Lee S.J.: Algorithmic solutions to decision problems in the braid groups, PhD Thesis, Korea Advanced Institute of Science and Technology (2000).

23. Lee S.J., Lee E.K.: Potential weakness of the commutator key agreement protocol based on braid groups, Proceedings of EUROCRYPT 2002, Lecture Notes in Computer Science, vol 2332, pp. 14–28. Springer-Verlag (2002).
24. Lee E., Park J.H.: Cryptanalysis of the public-key encryption based on braid groups, Advances in Cryptology: Proceedings of EUROCRYPT 2003, Lecture Notes in Computer Science, vol 2565, pp. 477–490. Springer-Verlag (2003).
25. Maffre S.: A weak key test for braid based cryptography. Des. Codes Cryptogr. **39**(3), 347–373 (2006).
26. Myasnikov A., Shpilrain V., Ushakov A.: A practical heuristic attack on the Ko-Lee key exchange protocol. Advances in Cryptology: Proceedings of CRYPTO 2005, Lecture Notes in Computer Science, vol 3621, pp. 86–96. Springer-Verlag (2005).
27. Sibert H.: Algorithmique des tresses, PhD Thesis, Universite de Caen (2003).
28. Sibert H., Dehornoy P., Girault M.: Entity authentication schemes using braid word reduction. Discrete Applied Math. **154**(2), 420–436 (2006).