POST QUANTUM GROUP THEORETIC CRYPTOGRAPHY

IRIS ANSHEL, DEREK ATKINS, DORIAN GOLDFELD, AND PAUL E. GUNNELLS

Abstract. Thanks to Shor's quantum factoring algorithm, the most prevalent asymmetric cryptographic systems (RSA, ECC) are now known to be vulnerable to attack by sufficiently powerful quantum computers. In this paper we discuss three Group Theoretic cryptographic protocols known as WalnutDSA (a digital signature algorithm), Hickory (a cryptographic hash function), and IronwoodKAP (a key agreement protocol), in the context of post-quantum cryptography. Unlike the classical public key protocols, the algebra underlying Walnut, Hickory, and Ironwood is non-abelian. We present evidence that these protocols are not susceptible to the quantum attacks known to be effective on RSA and ECC, and conclude that Group Theoretic Cryptography is a viable candidate for post-quantum cryptography.

1. Introduction

Secure communications have been around for millennia. As technology has improved and security needs have increased, developers of cryptographic protocols have responded by creating schemes based on advanced mathematical objects and techniques. The security of RSA, for instance, relies on the difficulty of factoring large integers. The security of Diffie–Hellman (DH) and Elliptic Curve Cryptography (ECC) relies on the difficulty of finding discrete logarithms in large finite cyclic groups. Although there are sophisticated attacks on these systems, no provably effective attacks are known.

The introduction of *quantum computing* into the picture has thrown a wrench into the works. Quantum computing provides algorithms that—when run on a sufficiently large quantum computer—can significantly affect the security of the cryptographic algorithms above. For instance, Shor's algorithm [30] can crack factoring or discrete logarithm problems; Grover's algorithm [21] can improve brute-force attacks by significantly reducing search spaces for private keys. As a result much current research has focused on cryptography that can survive into a post-quantum world.

In this paper, we discuss three Group Theoretic Public Key protocols: WalnutDSA (a digital signature scheme), IronwoodKAP (a key agreement protocol), and the Hickory Hash function. However, unlike the classic protocols, these group theoretic protocols are not based on finite cyclic abelian groups. Instead, the underlying algebraic object is the *braid group*, an infinite non-abelian group. The security of these group theoretic protocols is not based on any problem known to be susceptible to a quantum attack, which makes them viable candidates for post-quantum asymmetric cryptography. In addition, since the data transmitted over public channels is contained in the Braid group, linear algebra/permutation based attack are not applicable to the protocols presented here (see [22][23][4]).

Date: November 10, 2016.

1991 *Mathematics Subject Classification.* 20F36, 94A60.

Key words and phrases. Group theoretic cryptography, braid groups, digital signature, key agreement protocol, hash function, IoT, quantum resistant cryptography, security for the Internet of Things.

This paper is organized into nine sections. In Section 2 we give background on the braid group and the other algebraic structures used in our group theoretic protocols, and in Section 3 we describe our one-way function E-multiplication. Sections 4, 5, 6, introduce WalnutDSA, IronwoodKAP, and the Hickory Hash Function, respectively. The next two sections discuss the quantum resistance of these protocols. In Section 7 we discuss the impact of Grover's quantum search algorithm on the security of group theoretic cryptographic protocols, and Section 8 discusses quantum resistance of WalnutDSA, IronwoodKAP, and the Hickory Hash Function. Finally, in Section 9 we present our conclusions.

2. The Braid Group and Colored Burau matrices

Let B_N denote the N-strand braid group with Artin generators $\{b_1, b_2, \ldots, b_{N-1}\}$, subject to the following relations:

$$b_i b_{i+1} b_i = b_{i+1} b_i b_{i+1}, \qquad (i = 1, \ldots, N-2), \tag{1}$$

$$b_i b_j = b_j b_i, \qquad (|i-j| \geq 2). \tag{2}$$

Thus any $\beta \in B_N$ can be expressed as a product of the form

$$\beta = b_{i_1}^{\epsilon_1}\, b_{i_2}^{\epsilon_2}\, \cdots\, b_{i_k}^{\epsilon_k}, \tag{3}$$

where $i_j \in \{1, \ldots, N-1\}$, and $\epsilon_j \in \{\pm 1\}$.

Each braid $\beta \in B_N$ determines an element of S_N, the group of permutations of the set $\{1, \ldots, N\}$, as follows. For $1 \leq i \leq N-1$, let $\sigma_i \in S_N$ be the i^{th} simple transposition, which interchanges i and $i+1$ and leaves the remaining elements fixed. Then the map $b_i \mapsto \sigma_i$ defines a homomorphism $B_N \to S_N$. In particular, if $\beta \in B_N$ is written as in (3), then β is mapped to the permutation $\sigma_\beta = \sigma_{i_1} \cdots \sigma_{i_k}$.

Each braid generator b_i also determines an $N \times N$ matrix $CB(b_i)$, called the i^{th} *colored Burau matrix* [10]. Let $\{t_1, \ldots, t_N\}$ be a set of N indeterminates. For $i = 1$ and for $2 \leq i \leq N-1$, define $CB(b_i)$ by:

$$CB(b_1) = \begin{pmatrix} -t_1 & 1 & & & \\ & \ddots & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix}, \quad (4) \qquad CB(b_i) = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & t_i & -t_i & 1 & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix}. \quad (5)$$

Here the indicated variables appear in row i. We similarly define $CB(b_i^{-1})$ by modifying (4)–(5) slightly: we put

$$CB(b_i^{-1}) = \begin{pmatrix} 1 & & & & \\ & \ddots & & & \\ & 1 & -1/t_{i+1} & 1/t_{i+1} & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix},$$

where again the indicated variables appear in row i, and define $CB(b_1^{-1})$ by omitting the leftmost 1. For example, here are the colored Burau matrices in the case $N = 4$:

$$CB(b_1) = \begin{pmatrix} -t_1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \ CB(b_2) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ t_2 & -t_2 & 1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \ CB(b_3) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & t_3 & -t_3 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$CB(b_1^{-1}) = \begin{pmatrix} -1/t_2 & 1/t_2 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \ CB(b_2^{-1}) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & -1/t_3 & 1/t_3 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$CB(b_3^{-1}) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & -1/t_4 & 1/t_4 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

Recall that each generator b_i has an associated simple transposition σ_i. We may then associate to each generator b_i (respectively, inverse generator b_i^{-1}) a colored Burau/permutation pair $(CB(b_i), \sigma_i)$ (resp., $(CB(b_i^{-1}), \sigma_i)$). We now wish to define a multiplication of such colored Burau pairs so that the natural mapping from the braid group to the group of matrices with entries in the ring of Laurent polynomials in the t_i is a homomorphism. To accomplish this, we require the following observation. Given a Laurent polynomial $f(t_1, \ldots, t_N)$ in N variables, a permutation in $\sigma \in S_N$ can act on f from the left by permuting the indices of the variables. We denote this action by $f \mapsto {}^{\sigma}f$:

$${}^{\sigma}f(t_1, t_2, \ldots, t_N) = f(t_{\sigma(1)}, t_{\sigma(2)}, \ldots, t_{\sigma(N)}).$$

We extend this to an action of S_N on matrices over the ring of Laurent polynomials in the t_i by acting on each entry in the matrix, and similarly denote it by $M \mapsto {}^{\sigma}M$. The product of two colored Burau pairs is now defined as follows: given $b_i^{\pm}, b_j^{\pm}$, we put

$$(CB(b_i^{\pm}), \sigma_i) \circ (CB(b_j^{\pm}),\ \sigma_j) = \Big(CB(b_i^{\pm}) \cdot ({}^{\sigma_i}CB(b_j^{\pm})),\ \sigma_i \cdot \sigma_j\Big).$$

We extend this definition to the braid group inductively: given any braid

$$\beta = b_{i_1}^{\epsilon_1} b_{i_2}^{\epsilon_2} \cdots b_{i_k}^{\epsilon_k},$$

as in (3), we can define a colored Burau pair $(CB(\beta), \sigma_\beta)$ by

$$(CB(\beta), \sigma_\beta) = \\ (CB(b_{i_1}^{\epsilon_1}) \cdot {}^{\sigma_{i_1}}CB(b_{i_2}^{\epsilon_2}) \cdot {}^{\sigma_{i_1}\sigma_{i_2}}CB(b_{i_3}^{\epsilon_3})) \ \cdots \ {}^{\sigma_{i_1}\sigma_{i_2}\cdots\sigma_{i_{k-1}}}CB(b_{i_k}^{\epsilon_k}),\ \sigma_{i_1}\sigma_{i_2}\cdots\sigma_{i_k}).$$

One can check that this does indeed define a homomorphism from B_N to the group of matrices with Laurent polynomial entries.

As β varies over B_N, the entries of $CB(\beta)$ become Laurent polynomials of arbitrarily high degree. Thus computing $CB(\beta)$ for long braids β becomes very inefficient, even though the colored Burau matrices themselves are very simple. To overcome this problem and to make the colored Burau representation suitable for cryptographic applications, we introduce an additional step: E-multiplication.

3. E-multiplication$^{\text{TM}}$

The foundation of the Hickory Hash, IronwoodKAP, and WalnutDSA protocols, is based on a one-way function called *E-multiplication*. E-multiplication combines the theory of braids, the theory of matrices with polynomial entries (expressions of finite length constructed from variables), and modular arithmetic in small finite fields. Using E-multiplication, we can exchange the complexity of the colored Burau matrices into a non-cyclic, non-abelian finite set of matrix/permutation pairs and can reduce computation times.

Let q be a prime power, and let F_q be the finite field of q elements. A set of *T-values* is defined to be a collection of non-zero field elements:

$$\{\tau_1, \tau_2, \ldots, \tau_N\} \subset F_q.$$

Given a set of T-values, we can evaluate any Laurent polynomial $f(t_1, t_2, \ldots, t_N)$ to obtain an element of F_q:

$$f(t_1, t_2, \ldots, t_N) \downarrow_{\text{T-values}} := f(\tau_1, \tau_2, \ldots, \tau_N).$$

We extend this notation to matrices over Laurent polynomials in the obvious way.

With all these components in place we can now define E-multiplication. By definition, E-multiplication is an operation that takes as input two ordered pairs,

$$(M, \sigma_0), \quad (CB(\beta), \sigma_\beta),$$

where $\beta \in B_N$ and $\sigma_\beta \in S_N$ as before, and where $M \in GL(N, F_q)$, and $\sigma_0 \in S_N$. We denote E-multiplication with a star: $\star$. The result of E-multiplication, denoted

$$(M', \sigma') = (M, \sigma_0) \star (CB(\beta), \sigma_\beta),$$

will be another ordered pair $(M', \sigma') \in GL(N, F_q) \times S_N$.

We define E-multiplication inductively. When the braid $\beta = b_i^{\pm}$ is a single generator or its inverse, we put

$$(M, \sigma_0) \star \left(CB(b_i^{\pm}),\ \sigma_{b_i^{\pm}}\right) = \left(M \cdot {}^{\sigma_0}\!\left(CB(b_i^{\pm})\right) \downarrow_{\text{T-values}},\ \ \sigma_0 \cdot \sigma_{b_i^{\pm}}\right).$$

In the general case, when $\beta = b_{i_1}^{\epsilon_1} b_{i_2}^{\epsilon_2} \cdots b_{i_k}^{\epsilon_k}$, we put

$$(M, \sigma_0) \star (CB(\beta), \sigma_\beta) = (M, \sigma_0) \star (CB(b_{i_1}^{\epsilon_1}), \sigma_{b_{i_1}}) \star (CB(b_{i_2}^{\epsilon_2}), \sigma_{b_{i_2}}) \star \cdots \star (CB(b_{i_k}^{\epsilon_k}), \sigma_{b_{i_k}}), \quad (6)$$

where we interpret the right of (6) by associating left-to-right.) One can check that this is independent of the expression of β in the Artin generators.

Due to the sparseness of the CB matrices and the small sizes of the fields used in applications, E-Multiplication runs extremely fast, even in the smallest of computing devices including 8- and 16-bit microcontrollers.

4. WalnutDSA

Digital signatures provide a means for one party to create a document that can be sent through a second party and verified for integrity by a third party. This method ensures that the document was created by the first party and not modified by the second. Historically, digital signatures have been accomplished using various number-theoretic Public Key methods like RSA, DSA, and ECDSA.

WalnutDSA [13] is a new group theoretic digital signature protocol whose security is based on E-Multiplication, together with a new hard problem in braid groups, the cloaked conjugacy search problem, which appears immune to all the types of attacks related to the conjugacy search problem given in [14], [15], [17], [18], as well as the recent attacks in [7], [25], [27]. Due to the speed of E-Multiplication, Walnut allows for very rapid signature verification.

Definition (cloaking element) *Let $m \in GL(N, \mathbb{F}_q)$ and $\sigma \in S_N$. An element v in the pure braid subgroup of B_N is termed a cloaking element of (m, σ) if*

$$(m, \sigma) * (v, \mathrm{Id}_{S_N}) = (m, \sigma),$$

where Id_{S_N} is the identity permutation in S_N. The cloaking element is defined by the property that it essentially disappears when performing E-Multiplication.

Cloaked Conjugacy Search Problem (CCSP) *Consider the braid group B_N and symmetric group S_N with $N \geq 8$. Let $Y, v, v_1 \in B_N$ be unknowns where v cloaks (Id_N, Id_{S_N}) and v_1 cloaks Y. Assume $A \in B_N$ and $\mathcal{R}(Y^{-1}\, v\, A\, Y\, v_1)$ are known. Then it is infeasible to determine Y if the normal form of Y has a word length of at least 2000 Artin generators.*

The protocol for Walnut signature generation and verification proceeds as follows.

Digital Signature Generation:

1. Choose an integer $1 < i < n$.

2. Generate the cloaking elements v and v_1.

3. Generate the encoded message $E(\mathcal{M})$.

4. Compute $\mathrm{Sig} = \mathcal{R}\big(\mathrm{Priv(S)}^{-1} \cdot v \cdot E(\mathcal{M}) \cdot \mathrm{Priv(S)} \cdot v_1\big)$, which is a braid.

5. The final signature for the message $\mathcal{M}$ is the ordered pair $(\mathcal{M}, \mathrm{Sig})$.

Signature Verification: The signature $(\mathcal{M}, \mathrm{Sig})$ is verified as follows:

1. Generate the encoded message $E(\mathcal{M})$.

2. Define $\mathrm{Pub}(E(\mathcal{M}))$ by

$$\mathrm{Pub(E(\mathcal{M}))} \;=\; \big(\mathrm{Id}_N, \mathrm{Id}_{S_N}\big) \star E(\mathcal{M}).$$

where Id_N is the $N \times N$ identity matrix and Id_{S_N} is the identity permutation in S_N.

3. Evaluate the E-Multiplication $(\mathrm{M}_{PubSig}, \sigma_{PubSig}) = \mathrm{Pub(S)} \star \mathrm{Sig}$.

4. Verify the equality

$$\mathrm{MatrixPart}\big(\mathrm{Pub(S)} \star \mathrm{Sig}\big) = \mathrm{MatrixPart}\big(\mathrm{Pub(E(M))}\big) \cdot \mathrm{MatrixPart}\big(\mathrm{Pub(S)}\big),$$

where the matrix multiplication on the right is performed over the finite field. The signature is valid if this equality holds. If the results are not equal then the signature validation has failed.

5. IronwoodKAP

The Ironwood Key Agreement Protocol [19] allows two devices, each of which is in possession of some public system data and some private data, to obtain a shared secret after an exchange of some public data (public keys) over an open communication channel. The first device may be enabled to evaluate E-Multiplication using selectively pre-shared data, which the second device does not require. The first device may authenticate its public key by transmitting a signed certificate along with its public key. The public key of the second devise may not require authentication.

The system data for the IronwoodKAP consists of a group S acting on the left on a monoid M, and a second group N which is also a vector space, and abelian subgroup $N_0 \leq N$. The case of the monoid M being the colored Burau representation of the Braid group, the group S being the symmetric group, and the group N being $GL(N, F_q)$ can be implemented very efficiently.

First Device Private Data:

• E-Multiplication Data: A homomorphism $\Pi : M \to N$, which enables the first device to evaluate the one–way function E- multiplication $\star$, together with two E-commuting submonoids $A, B \leq M \rtimes S$.

• Two secret elements $(m_1, s), (m_2, s) \in A$ and two secret elements $c_1, c_2 \in N_0$.

• Using the secret elements, the first device evaluates two ordered pairs associated with its private keys, $(c_1 \cdot \Pi(m_1), s)$, $(c_2 \cdot \Pi(m_2), s) \; \in \; N \times S$.

First Device Public Data:

• The first device public key is a combination of $(c_1 \cdot \Pi(m_1)) \cdot (c_2 \cdot \Pi(m_2))^{-1}$, generated above, and a value generated by the first device upon receiving the public key from the second device. The first device generates the public key data, using E-multiplication, in the manner described in Steps 1–4 of the protocol below.

Second Device Private Data:

An element $c_0 \in N_0$.

Second Device Public Data:

• The Ironwood public key $(c_0 \cdot \Pi(m_0), s_0)$, which is evaluated by the TTP, where (m_0, s_0) is a fixed element in the submonoid B that is specified for the second device.

Ironwood Key Agreement Protocol:

STEP 1: The second device sends its Ironwood public key over an open channel to the first device.

STEP 2: The first device uses the Ironwood public key of the second device and each of its private keys to compute:

$$(c_1, 1) \cdot (c_0 \cdot \Pi(m_0) \star (m_1, s) = (c_0, 1)(c_1 \cdot \Pi(m_1), s) \star (m_0, s_0) = (Y_1, s_0 s),$$

$$(c_2, 1) \cdot (c_0 \cdot \Pi(m_0) \star (m_2, s) = (c_0, 1)(c_2 \cdot \Pi(m_2), s) \star (m_0, s_0) = (Y_2, s_0 s).$$

STEP 3: The first device applies a projection operator proj:$N \to$ a lower dimensional subspace of N to the elements Y_1, Y_2 to obtain $V_1 = \text{proj}(Y_1),\ V_2 = \text{proj}(Y_2)$.

STEP 4: The first device sends the session public key $\Big((c_1 \cdot \Pi(m_1)) \cdot (c_2 \cdot \Pi(m_2))^{-1},\ V_1 \Big)$.

STEP 5: The second device evaluates $V_2 = c_0 \cdot (c_1 \cdot \Pi(m_1) \cdot (c_2 \cdot \Pi(m_2))^{-1} \cdot c_0^{-1}$.

STEP 6: Both devices are now in possession of the value V_2. When both devices apply a cryptographic hash function to V_2, the output is the shared secret produced by the Ironwood-KAP.

6. Hickory Hash Function

The Hickory Hash Function is based on a protocol that first appeared in 2016 [2]. In brief, it is a family of hash functions specified by the following data:

$$\big\{B_N,\ \ q,\ \ \lambda,\ \ t\text{-values} = \{\tau_1, \ldots, \tau_N\},\ \ \{c_0, c_1, \ldots, c_{2^\lambda-1}\} \subset B_N,\ \ (n_0, \sigma_0) \in N_q \times S_N\big\},$$

where

- B_N is the braid group on N strands;
- q is a power of 2, the t-values are invertible elements in F_q;
- the collection of braid group elements $\{c_0, c_1, \ldots, c_{2^\lambda-1}\}$ is fixed and assumed to generate a free submonoid of B_N;
- $(n_0, \sigma_0) \in N_q \times S_N$ is an ordered pair.

The output of the Hash function is defined to be the sequence of bits that specify the matrix, which is evaluated through a sequence of E-multiplications where the t-values change at each step. The length of the Hash is given by

$$N^2 \cdot \text{ceil}\big(\log_2(q)\big),$$

where for $x > 0$, the function ceil(x) (denotes the ceiling of x) which is the smallest integer n such that $x \leq n$.

The Hickory Hash Function is a specific instantiation of this Hash family, specifying all the parameter data, defined in [3].

7. Quantum resistance of WalnutDSA, IronwoodKAP, Hickory Hash I: Grover's quantum search algorithm

A cryptographic protocol is said to be *quantum resistant* if it remains secure even when an attacker has access to a quantum computer and can perform polynomial time quantum computations. In this section and the next, we present evidence that WalnutDSA, IronwoodKAP, and Hickory Hash are viable quantum resistant protocols.

Grover's quantum search algorithm [21] allows a quantum computer to search for a particular item in an unordered n-element search space in $\mathcal{O}\left(\sqrt{n}\right)$ steps. By comparison, searching on a classical computer takes $\mathcal{O}\left(n\right)$ steps. It follows that if the security level of a cryptosystem is based on a brute force search of a keyspace, then Grover's quantum search algorithm will reduce a security level of 2^k to $2^{k/2}$. If the running time of the cryptosystem is quadratic in the security level, then Grover's algorithm will generally result in a fourfold increase in running time.

Now the running time of WalnutDSA, IronwoodKAP, and Hickory Hash, is linear in the security level. In order to double the security strength of these protocols, the complexity runtime doubles. This means that if an attacker has access to a quantum computer, then the running time will only have to double to maintain the same security level because doubling the security level (to counteract Grover's algorithm) only requires double the runtime.

8. Quantum resistance of WalnutDSA, IronwoodKAP, Hickory Hash II: Shor's algorithm and the hidden subgroup problem

It was observed by Kitaev [24] that the quantum algorithms that have been developed to break classical cryptographic systems such as those based on the hardness of computing discrete logarithms [30] can be vastly generalized to solve the *Hidden Subgroup Problem* (HSP):

Hidden Subgroup Problem: Let G be a finite group and let H be a subgroup of G. For a finite set X, let $f\colon G \to X$ be a function that is constant and distinct on left cosets of H. In other words $f(gh) = f(gh')$ for any fixed $g \in G$ and all $h, h' \in H$ and $f(gh) \neq f(g'h)$ for $g \neq g'$ $(g, g' \in G)$ and any $h \in H$. We say f *hides* the subgroup $H \subset G$. The *Hidden Subgroup Problem* is then the following: *given an f as above, find the subgroup H that it hides.*

Shor's algorithm [30] reduces the hard problem of factoring to finding the order of an element in a cyclic group (discrete logarithm problem). He then solves the order finding problem with a quantum period-finding subroutine that reduces to the HSP for cyclic groups. It is known that the HSP can be solved on a quantum computer when the hidden subgroup H is abelian [26].

By contrast, at present there is no known polynomial-time solution to the HSP when the hidden subgroup is non-abelian, although some special cases have been solved [5, 8, 12, 16, 20].

The WalnutDSA, IronwoodKAP, and Hickory Hash protocols take place on the braid group B_N, which is an infinite non-abelian group. The security of these protocols is based on the hardness of reversing E-mulitplication and CCSP. There seems to be no way to connect these hard problems with HSP.

The core operation in WalnutDSA, IronwoodKAP, and Hickory Hash is E-multiplication, as described in Section 3. Given an element

$$\beta = b_{i_1}^{\epsilon_1}\, b_{i_2}^{\epsilon_2}\, \cdots\, b_{i_k}^{\epsilon_k} \in B_N, \tag{7}$$

where $i_j \in \{1, \dots, N-1\}$, and $\epsilon_j \in \{\pm 1\}$, we can define a function $f : B_N \to GL(N, F_q)$ where $f(\beta)$ is given by the E-multiplication $(1,1) * (\beta, \sigma_\beta)$ and σ_β is the permutation associated to β. Now E-multiplication is a highly non linear operation. As the length k of the word β increases, the complexity of the Laurent polynomials occurring in the E-multiplication defining $f(\beta)$ increases exponentially. It does not seem to be possible that the function f exhibits any type of simple periodicity, so it is very unlikely that inverting f can be achieved with a polynomial quantum algorithm. Note that this does reduce the group theoretic protocols discussed in this paper into a finite, yet still highly non-cyclic, non-abelian group; however this does not affect the analysis.

The class NP (nondeterministic polynomial time) is the set of all decision problems with the property that if someone reveals the answer to the decision problem, then it is possible to vary the answer in polynomial time. In [6] it is shown that relative to an oracle chosen uniformly at random with probability 1, the class NP cannot be solved on a quantum Turing machine in time $o(2^{n/2})$. In this sense the quantum search algorithms which are purely based on the blackbox property of the formulae, cannot solve the search problem in time $o(2^{n/2})$. This implies that the search problem remains resistant in the quantum computational model.

Given a group presentation G and a word w in G, consider the class of all words in G that are equivalent to w, under the defining relations of G. Can we find the element in this class of words which has the shortest word length in the generators and their inverses? This is called the *shortest word problem* in the group G. Under the assumption that the shortest word problem in the braid group can be efficiently solved [27] developed an heuristic length attack on AEDH which can be refuted for large key lengths [23]. In 1991, Paterson–Razborov [28] showed that the shortest word problem in the braid group on infinitely many strands is at least as hard as an NP-complete problem. This suggests that the shortest word problem in the braid group on infinitely many strands may be quantum resistant. Tatsuoka [31] has shown that the shortest word problem in the braid group on a fixed finite number of strands can be solved in quadratic time. But the constants in the estimate of the running time must be growing exponentially large as the number of strands increase for the Paterson–Razborov result to hold. This lends further credibility for the choice of group theoretic protocols described in this paper as a viable choice for post quantum cryptography.

9. Conclusions

At its core, WalnutDSA, IronwoodKAP, and Hickory Hash utilize the infinite non-abelian braid group in concert with a distinct one way function (E-multiplication) which is both rapidly computable and erases the data required for reversal.

A central feature to E-multiplication based protocols is that the running time and security level grow linearly with the length of the private key. This stands in stark contrast to other Public Key systems which grow quadratically. This feature ensures that a quantum search attack on E-multiplication based protocols, is that the running time and security level will only require doubling the running time of the algorithm (instead of increasing it by a factor of 4) while maintaining the same security level as in a pre-quantum world.

Shor's algorithm [30] has made cryptographic protocols whose security is based on the hardness of computing discrete logarithms vulnerable to quantum attacks. Shor's algorithm has been further generalized to a larger set of hard problems now collectively known as the *Hidden Subgroup Problem* (HSP). The HSP has now been shown to be solvable on a quantum computer when the hidden subgroup is finite abelian or is one of a small class of finite non-abelian groups. The braid group does not contain any non-trivial finite subgroups at all, making it a viable quantum-resistant candidate for cryptography.

References

[1] Anshel, Iris; Anshel, Michael; Goldfeld, Dorian; and Lemieux, Stephane, *Key agreement, the Algebraic Eraser^{TM}, and Lightweight Cryptography,* Algebraic methods in cryptography, Contemp. Math., vol. 418, Amer. Math. Soc., Providence, RI, 2006, pp. 1–34.

[2] Anshel, Iris; Atkins, Derek; Goldfeld, Dorian; Gunnells, Paul E.; *A Class of Hash Functions Based on the Algebraic Eraser,* Groups Complex. Cryptol. 2016; 8 (1):1–7.

[3] Anshel, Iris; Atkins, Derek; Goldfeld, Dorian; Gunnells, Paul E.; *Hickory Hash$^{\text{TM}}$: Implementing an Instance of an Algebraic Eraser$^{\text{TM}}$ Hash Function on an MSP430 Microcontroller,* Publication forthcoming, 2016.

[4] Anshel, Iris; Atkins, Derek; Goldfeld, Dorian; Gunnels, Paul E.; *Defeating the Ben-Zvi, Blackburn, and Tsaban Attack on the Algebraic Eraser,* 2016, arXiv:1601.04780

[5] D. Bacon, A. Childs, and W. van Dam; *From optimal measurements to efficient quantum algorithms for the hidden subgroup problem over semidirect product groups,* 46th Annual IEEE Symposium on Foundations of Computer Science, pages 469–478, 2005.

[6] Charles H. Bennett, Ethan Bernstein, Gilles Brassard, and Umesh Vazirani; *Strengths and weaknesses of quantum computing,* SIAM J. Comput., 26(5):1510–1523, October 1997.

[7] Adi Ben-Zvi, Simon R. Blackburn and Boaz Tsaban, *A practical cryptanalysis of the Algebraic Eraser*, Advances in Cryptology – CRYPTO 2016, to appear. See http://eprint.iacr.org/2015/1102.

[8] T. Beth and M. Rötteler; *Polynomial-time solution to the hidden subgroup problem for a class of non-abelian groups,* 1998, arXiv:9812070

[9] Birman, Joan; Ko, Ki Hyoung; Lee, Sang Jin; *A new approach to the word and conjugacy problems in the braid groups,* Adv. Math. 139 (1998), no. 2, 322–353.

[10] H.R. Morton, *The multivariable Alexander polynomial for a closed braid, Low-dimensional topology,* (Funchal, 1998), 167–172, Contemp. Math., 233, Amer. Math. Soc., Providence, RI, 1999.

[11] Dehornoy, Patrick; *A fast method for comparing braids,* Adv. Math. 125 (1997), no. 2, 200–235.

[12] K. Friedl, G. Ivanyos, F. Magniez, M. Santha, and P. Sen; Hidden translation and orbit coset in quantum computing. In Proc. 35th Annual ACM Symposium on Theory of Computing, pages 1–9, 2001.

[13] *Algebraic Eraser Digital Signature System,* Provisional Patent, September, 2015.

[14] Garber, David; Kaplan, Shmuel; Teicher, Mina; Tsaban, Boaz; Vishne, Uzi, *Length-based conjugacy search in the braid group,* Algebraic methods in cryptography, 75-87, Contemp. Math., 418, Amer. Math. Soc., Providence, RI, 2006.

[15] V. Gebhardt, *A new approach to the conjugacy problem in Garside groups,*, J. Algebra 292(1) (2005), 282–302.

[16] D. Gavinsky; *Quantum solution to the hidden subgroup problem for poly-near- hamiltonian groups,* Quantum Information and Computing, 4:229–235, 2004.

[17] Anja Groch; Dennis Hofheinz; and Rainer Steinwandt, *A Practical Attack on the Root Problem in Braid Groups,* Algebraic methods in cryptography, 121-131, Contemp. Math., 418, Amer. Math. Soc., Providence, RI, 2006.

[18] Dennis Hofheinz; Rainer Steinwandt, *A practical attack on some braid group based cryptographic primitives,* Public Key Cryptography, Proceedings of PKC 2003 (Yvo Desmedt, ed.), Lecture Notes in Computer Science, no. 2567, Springer-Verlag, 2002, pp. 187-198.

[19] *Shared Secret Data Production System,* Provisional Patent, May, 2016.

[20] Grigni, M., Schulman, L., Vazirani, M., and Vazirani, U.; *Quantum mechanical algorithms for the non-abelian hidden subgroup problem,* in Proc. 33rd Annual ACM Symposium on Theory of Computing, page 6874, 2001.

[21] Grover L.K.; *A fast quantum mechanical algorithm for database search,* Proceedings, 28th Annual ACM Symposium on the Theory of Computing, (May 1996) p. 212.

[22] Gunnells, Paul; *Defeating the Kalka–Teicher–Tsaban linear algebra attack on the Algebraic Eraser,* 2012, arXiv:1202.0598

[23] Gunnells, Paul; *On the cryptanalysis of the generalized simultaneous conjugacy search problem and the security of the Algebraic Eraser,* 2011, arXiv:1105.1141

[24] Kitaev, Alexi Yu,; *Quantum measurements and the Abelian stabilizer problem,* 1995, quant-ph/9511026 .

[25] A. Kalka, M. Teicher, and B. Tsaban,; *Cryptanalysis of the Algebraic Eraser and short expressions of permutations as products,* 2008, arXiv:0804.0629v4.

[26] Lomont, C.; *The hidden subgroup problem - review and open problems,* 2004, arXiv:0411037

[27] A. D. Myasnikov and A. Ushakov,;*Cryptanalysis of the Anshel-Anshel-Goldfeld-Lemieux key agreement protocol,* Groups Complex. Cryptol. 1 (2009), no. 1, 63–75.

[28] M.S. Paterson and A.A. Razborov, *The Set of Minimal Braids is co-NP-Complete,* J. Algorithms,12, (1991), 393–408.

[29] D. Atkins, *Algebraic Eraser: A lightweight, efficient asymmetric key agreement protocol for use in no-power, low-power, and IoT devices,* NIST Lightweight Cryptography Workshop, 2015.

[30] Shor, Peter; *Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer,* SIAM J. on Computing, (1997) 1484–1509.

[31] Tatsuoka, Kay; *An isoperimetric inequality for the Artin groups of finite type,* Trans. of the Amer. MBest,ath. Soc., Volume 339, Number 2 (1993), 537–551.

SecureRF Corporation, 100 Beard Sawmill Rd #350, Shelton, CT 06484
E-mail address: ianshel@securerf.com

SecureRF Corporation, 100 Beard Sawmill Rd #350, Shelton, CT 06484
E-mail address: datkins@securerf.com

SecureRF Corporation, 100 Beard Sawmill Rd #350, Shelton, CT 06484
E-mail address: dgoldfeld@securerf.com

SecureRF Corporation, 100 Beard Sawmill Rd #350, Shelton, CT 06484
E-mail address: pgunnells@securerf.com