A Transformation for Lifting Discrete Logarithm Based Cryptography to Post-Quantum Cryptography

Danilo Gligoroski*

March 3, 2023

Abstract

We construct algebraic structures where rising to the non-associative power indices is no longer tied with the Discrete Logarithm Problem but with a problem that has been analysed in the last two decades and does not have a quantum polynomial algorithm that solves it. The problem is called Exponential Congruences Problem. By this, *we disprove* the claims presented in the ePrint report 2021/583 titled "Entropoids: Groups in Disguise" by Lorenz Panny that *"all instantiations of the entropoid framework should be breakable in polynomial time on a quantum computer."*

Additionally, we construct an Arithmetic for power indices and propose generic recipe guidelines that we call "Entropic-Lift" for transforming some of the existing classical cryptographic schemes that depend on the hardness of Discrete Logarithm Problem to post-quantum cryptographic schemes that will base their security on the hardness of the Exponential Congruences Problem.

As concrete examples, we show how to transform the classical Diffie-Hellman key exchange, DSA and Schnorr signature schemes.

We also post one open problem: From the perspective of provable security, specifically from the standpoint of security of post-quantum cryptographic schemes, to precisely formalize and analyze the potentials and limits of the Entropic-Lift transformation.

1 Mathematical preliminaries

We give here some basic definitions and prove some properties about the algebraic structures that we will use. For more definitions reader can consult any standard textbook on abstract algebra topics.

In further text, using multiplicative notation, we will assume that $(G, \cdot)$ is a finite commutative group with operation $\cdot$ and a unit element 1_G. Further, we will assume G has $|G| = q^2$ elements. We will also assume that q can be represented as a product of ν distinct prime factors $q = q_1 \ldots q_\nu$, (sorted in ascending order), where the smallest prime factor $q_1 > 2$. In general case the number of prime factors ν can be arbitrary, but for cryptographic purposes we will assume that it is as small as possible, and the biggest factor q_ν is as big as possible (for example $q_\nu > q^{1/2}$). Thus, we will work with a group G of order $|G| = q_1^2 \ldots q_\nu^2$. We will use a short notation $[\nu]$ to annotate the set $\{1, \ldots \nu\}$.

Next assumption about the group G is that it is not cyclic, but is generated with two independent elements $g_1, g_2 \in G$, i.e. $\forall x \in G$, $\exists i, j \in \mathbb{Z}_q$, s.t. $x = g_1^i \cdot g_2^j$.

With other words we take that G is a direct product of two maximal cyclic subgroups $G_1 = \langle g_1 \rangle$ and $G_2 = \langle g_2 \rangle$ i.e.

$$G \cong G_1 \times G_2,$$

where the order of G_1 and G_2 is q i.e.

$$|\langle g_1 \rangle| = |\langle g_2 \rangle| = q.$$

From the properties of abelian groups, we have the following

Proposition 1. *For every $x \in G$, there are unique $i, j \in \mathbb{Z}_q$, s.t. $x = g_1^i \cdot g_2^j$.*

Definition 1. *An automorphism α on the group $(G, \cdot)$ is a bijective homomorphism of G to itself i.e.*

- *$\alpha : G \mapsto G$ is a bijection;*

*Department of Information Security and Communication Technologies, Norwegian University of Science and Technology - NTNU

- *α is homomorphism with the respect of the operation $\cdot$ i.e.*

$$\forall x, y \in G, \alpha(x \cdot y) = \alpha(x) \cdot \alpha(y).$$

Definition 2. *An involutive automorphism T on the group $(G, \cdot)$ is an automorphism that is also an involution i.e.*

$$T : G \mapsto G \text{ is bijection,}$$
$$\forall x, y \in G, \quad T(x \cdot y) = T(x) \cdot T(y), \text{ and}$$
$$\forall x \in G, \quad T(T(x)) = T^{(2)}(x) = x.$$

From T being automorphism, the following property holds:

Corollary 1. *For all $x \in G$,*

$$(T(x))^j = T(x^j).$$

Proposition 2. *If $h \in G$ is an element such that h and $T(h)$ are independent elements of order q, then for every $x \in G$ there is a unique pair $(i, j) \in \mathbb{Z}_q \times \mathbb{Z}_q$ such that*

$$x = h^i \cdot T(h^j). \tag{1}$$

Proof. Let us denote by $H_1 = \langle h \rangle$ the subgroup generated by h and $H_2 = \langle T(h) \rangle$ the subgroup generated by $T(h)$. Having h and $T(h)$ being independent means that $H_1 \cap H_2 = \{1_G\}$, $|H_1| = |H_2| = q$ and that $G \cong H_1 \times H_2$. Then the uniqueness of the pair of indices (i, j) follows from the Proposition 1. □

Definition 3. *For every $x, y \in G$, let us define a non-commutative binary operation*

$$x \boxplus y \equiv x \cdot T(y).$$

We call the algebraic structure $\mathbb{E}_{q^2} = (G, \boxplus, \cdot)$ a finite Entropoid with q^2 elements.

We collect several properties of the operation $\boxplus$ in the following Proposition.

Proposition 3.

- *For $x, y \in G$, in general, $x \boxplus y \neq y \boxplus x$;*
- *The multiplicative unit 1_G acts as a a right zero in $(G, \boxplus)$ i.e. for all $x \in G$, $x \boxplus 1_G = x$;*
- *From the left, 1_G acts as the involution T on x i.e. for all $x \in G$, $1_G \boxplus x = T(x)$;*
- *To define a consistent operation $\boxminus$ that will act as "the opposite" operation to $\boxplus$ in $(G, \boxplus)$ we need to define it as: For all $x, y \in G$, $x \boxminus y \overset{def}{=} x \cdot T(y^{-1}) = x \boxplus (y^{-1})$. In that case, if $x \boxminus y = z$, then $x = z \boxplus y$.*

Definition 4. *We call an element $g \in G$* a generator of the Entropoid $\mathbb{E}_{q^2}$ *if g and $T(g)$ are independent elements of order q.*

We adapt Algorithm 4.80 from [9] that finds a generator of a cyclic group for finding a generator of an Entropoid.

Lemma 1. *If g is an output of Algorithm defined in Table 1, then it is a generator for the Entropoid $\mathbb{E}_{q^2}$.*

Proof. Since $G \cong G_1 \times G_2$ where the order of G_1 and G_2 is q the computation of the set $\mathcal{B}$ in Step 2, and the check in Step 3 if 1_G belongs to $\mathcal{B}$ ensure that g is a generator of a maximal cyclic subgroup of order q i.e. $|\langle g \rangle| = q$. Similar reasoning applies for the set $\mathcal{B}_T$, that ensures $T(g)$ is a generator of a maximal cyclic subgroup of order q i.e. $|\langle T(g) \rangle| = q$.

The final check in Step 4 checks if g and $T(g)$ are independent elements which from Definition 4 is the necessary condition for g to be a generator for the Entropoid $\mathbb{E}_{q^2}$. □

```
Algorithm: Finding a generator of an Entropoid
Input: An Entropoid $\mathbb{E}_{q^2}$, and the prime factorization $q^2 = q_1^2 \dots q_\nu^2$
Output: Generator $g$.

  1. Choose a random element $g \in G$
  2. Compute the sets
     $\mathcal{B} = \{b \mid b = g^{(q^2/q_i)}$, for $i \in [\nu]\}$ and
     $\mathcal{B}_T = \{b \mid b = T(g)^{(q^2/q_i)}$, for $i \in [\nu]\}$.
  3. If $1_G \in \mathcal{B}$ or $1_G \in \mathcal{B}_T$ then go to Step 1.
  4. If $\mathcal{B} \cap \mathcal{B}_T \neq \emptyset$ then go to Step 1.
  5. Return $g$.
```

Table 1: Finding a generator of and Entropoid

Definition 5. *Let the two-dimensional exponents $X = (x_1, x_2)$ be called* power indices. *For every $g \in G$ we define exponentiation with the power index $X = (x_1, x_2)$ as:*

$$g^X = g^{(x_1,x_2)} = g^{x_1} \cdot T(g^{x_2}).$$

Lemma 2 (Arithmetic of power indices)**.** *Let $X = (x_1, x_2)$ and $Y = (y_1, y_2)$ be two power indices, and let us define the following operations (where mod q acts component-wise):*

addition: $X + Y \equiv (x_1, x_2) + (y_1, y_2) = ((x_1 + y_1),\ (x_2 + y_2)) \mod q$

subtraction: $X - Y \equiv (x_1, x_2) - (y_1, y_2) = ((x_1 - y_1),\ (x_2 - y_2)) \mod q$

multiplication: $XY \equiv (x_1, x_2) \times (y_1, y_2) = ((x_1y_1 + x_2y_2),\ (x_1y_2 + x_2y_1)) \mod q$

division: $\frac{X}{Y} \equiv \frac{(x_1,x_2)}{(y_1,y_2)} = \left(\frac{x_1y_1 - x_2y_2}{y_1^2 - y_2^2},\ \frac{x_2y_1 - x_1y_2}{y_1^2 - y_2^2}\right) \mod q$, *assuming the following condition* $\mathrm{GCD}(y_1^2 - y_2^2, q) = 1$

Then, for every $h \in G$ the following relations hold:

$$h^X \cdot h^Y = h^{(X+Y)},$$

$$h^X \cdot (h^Y)^{-1} = h^{(X-Y)},$$

$$(h^X)^Y = h^{(XY)}.$$

For the division operation the following computational problem is solved: Given, h, X and Y find a power index Z such that

$$h^Z = h_1 \quad \textit{and} \quad h^X = h_1^Y.$$

Proof. While it may seem that the arithmetic expressions are complicated, their consistency can be checked with simple manipulations with algebraic expressions and taking in consideration Definition 2 and Corollary 1. □

Lemma 3. *Let the power index $S = (s_1, s_2)$ is such that* $\mathrm{GCD}(s_1^2 - s_2^2, q) = 1$. *Then the mapping $\sigma : G \mapsto G$ defined as $\sigma(x) = x^S$ is an automorphism of G.*

Proof. The mapping σ is homomorphism since for all $x, y \in G$ we have $\sigma(x \cdot y) = (x \cdot y)^S = (x \cdot y)^{s_1} \cdot T((x \cdot y)^{s_2}) = x^{s_1} \cdot y^{s_1} \cdot T(x^{s_2} \cdot y^{s_2}) = x^{s_1} \cdot y^{s_1} \cdot T(x^{s_2}) \cdot T(y^{s_2}) = x^{s_1} \cdot T(x^{s_2}) \cdot y^{s_1} \cdot T(y^{s_2}) = \sigma(x) \cdot \sigma(y)$.

We prove that σ is injection as follows. Let $\sigma(x_1) = \sigma(x_2)$ i.e. $x_1^{(s_1,s_2)} = x_2^{(s_1,s_2)}$. Let us set $h = \frac{x_1}{x_2} = x_1 \cdot x_2^{-1}$ and $X = (1, 0)$. From the condition that $\mathrm{GCD}(s_1^2 - s_2^2, q) = 1$ it follows that there is a power index $Z = \frac{X}{S} = \left(\frac{s_1}{s_1^2 - s_2^2}, \frac{-s_2}{s_1^2 - s_2^2}\right)$, such that $h_1 = h^Z$ and $h^X = h_1^S$. Computing fist h_1 we have: $h_1 = \left(\frac{x_1}{x_2}\right)^Z = \left(\frac{x_1}{x_2}\right)^{\frac{s_1}{s_1^2 - s_2^2}} \cdot T\left(\left(\frac{x_1}{x_2}\right)^{\frac{-s_2}{s_1^2 - s_2^2}}\right)$. Then replacing h and h_1 in $h^X = h_1^S$ we have:

$$\left(\frac{x_1}{x_2}\right)^{(1,0)} = \left(\left(\frac{x_1}{x_2}\right)^{\frac{s_1}{s_1^2 - s_2^2}} \cdot T\left(\left(\frac{x_1}{x_2}\right)^{\frac{-s_2}{s_1^2 - s_2^2}}\right)\right)^{(s_1, s_2)}$$

which is equivalent to

$$\left(\frac{x_1}{x_2}\right)^{(1,0)} = \left(\left(\frac{x_1}{x_2}\right) \cdot T\left(\left(\frac{x_1}{x_2}\right)\right)\right)^{(1,0)}$$

which further reduces to

$$\frac{x_1}{x_2} = \frac{x_1}{x_2} \cdot T\left(\frac{x_1}{x_2}\right)$$

and further to

$$1_G = T\left(\frac{x_1}{x_2}\right).$$

Since T is involutive automorphism, $T(1_G) = 1_G$, and by applying T on both sides of the last equality we have

$$1_G = \frac{x_1}{x_2},$$

i.e. $x_1 = x_2$. □

In its most general form, Exponential Congruences Problem (ECP) is seeking for a solution of the equation

$$a g_1^{x_1} + b g_2^{x_2} = c \tag{2}$$

in an algebraic structure defined over two operations $(G, +, *)$, with q elements i.e., $|G| = q$, where $a, b, g_1, g_2, c \in G$, g_1 and g_2 have respectively orders s and t i.e., $|\langle g_1 \rangle| = s$ and $|\langle g_2 \rangle| = t$, and where $s, t < q$ but $st \geq q$ [19, 21].

Definition 6. *Let g be the generator of the Entropoid $\mathbb{E}_{q^2}$, and let the DLP be a computationally hard problem over the cyclic subgroup $(G_1, \cdot)$ generated by g i.e. $G_1 = \langle g \rangle$. With other words for a given $y \in G_1$, where $y = g^x$ we assume that there is no (classical) polynomial time algorithm that finds $x \in \mathbb{Z}_q$.* ***Entropic-Lift*** *of the DLP is the transformation that replaces the exponents $x \in \mathbb{Z}_q$ with two-dimensional power indices $X = (x_1, x_2)$ where $x_1, x_2 \in \mathbb{Z}_q$. More concretely, for a given $y \in G$, where $y = g^X$ find the power index $X = (x_1, x_2)$.*

Lemma 4. *The elevated DLP is a simplified Exponential Congruence Problem in the Entropoid $\mathbb{E}_{q^2}$, where $a = b = 1$ and $g_1 = g_2 = g$ i.e. has the following form:*

$$y = g^{x_1} \boxplus g^{x_2} \tag{3}$$

Proof. The elevated DLP is the following problem: for a given $y \in G$ it seeks to find $x_1, x_2 \in \mathbb{Z}_q$ s.t. $y = g^{(x_1, x_2)}$. Directly from the definition of exponentiation with power indices in $\mathbb{E}_{q^2}$ we have that $y = g^{x_1} \boxplus g^{x_2}$. The most important part is the fact that g and $T(g)$ are independent, which prevents neither g nor $T(g)$ to be represented as powers of each other. That prevents a collapse of the two-dimensionality of the power indices to a one-dimensional case, which would be the Discrete Logarithm Problem. The obtained variant of the Exponential Congruence expression is indeed a simplified variant of the general ECP where $a = b = 1$ and $g_1 = g_2 = g$. □

Recipe for Entropic-Lift of a cryptographic scheme Let g be the generator of the Entropoid $\mathbb{E}_{q^2}$, and let $\mathcal{S}(g, \mathcal{A})$ is a cryptographic scheme with a set of algorithms $\mathcal{A} = \{\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_\nu\}$ that bases its security on DLP. Let DLP be a computationally hard problem over the cyclic subgroup $(G_1, \cdot)$ generated by g. Let the set of algorithms $\mathcal{A}$ use in total μ exponent variables denoted with $x^{(i)} \in \mathbb{Z}_q$, where $i \in \{1, \ldots \mu\}$. **Entropic-Lift** of the scheme $\mathcal{S}$ is the transformation that replaces all used exponent variables $x^{(i)} \in \mathbb{Z}_q$ in algorithms $\mathcal{A}$, with two-dimensional power indices $X^{(i)} = (x_1^{(i)}, x_2^{(i)})$, $i \in \{1, \ldots \mu\}$. The replacements include all expressions: exponential expressions and arithmetic expressions with ordinary indices from $\mathbb{Z}_q$. A special attention should be devoted to the following situation: If the lifted scheme needs to perform an operation between a power index and a group element variable, then that operation is not defined with the arithmetic of power indices. In that case try to map the group element variable by a cryptographic hash function to a power index variable in order to make the operation possible. Re-evaluate the security of the lifted scheme.

2 The rationale for using involutive automorphisms $T : G \mapsto G$

The first attempt to use entropic non-commutative and non-associative quasigroups [4] was cryptanalyzed by Panny in [13].

The design idea was to define a general class of groupoids $(G, *)$ (sets G with a binary operation $*$ that is both non-commutative and non-associative) that are *"Entropic"* i.e. for every four elements x, y, z and w, a pseudo-associativity law is satisfied:

$$(x * y) * (z * w) = (x * z) * (y * w).$$

In order to compute the powers x^a where $a \in \mathbb{Z}^+$, of elements $x \in G$, due to the non-associativity, the exact bracketing shape a_s is also required to be known, and Etherington called those general exponentiation indices *power indices*. They can be denoted as pairs $\mathbf{A} = (a, a_s)$. Further, for those entropic groupoids Etherington showed that they satisfy the *"Palintropic"* property i.e., $x^{\mathbf{AB}} = (x^{\mathbf{A}})^{\mathbf{B}} = (x^{\mathbf{B}})^{\mathbf{A}} = x^{\mathbf{BA}}$. Those relations are exactly the Diffie-Hellman key exchange protocol relations used with groups. The work [4] proposed a succinct notation of the exponentially large non-associative power indices. However, the instances proposed there and later in [5] were successfully cryptanalyzed by Panny in [13].

Panny intelligently used a theorem proved by Murdoch [11], Toyoda [20] and Bruck [2]:

Theorem 1 (Theorem 1 in [13])**.** *For every entropic quasigroup $(G, *)$, there exist an abelian group $(G, \cdot)$, commutative automorphisms σ, τ of $(G, \cdot)$, and an element $c \in G$, such that*

$$x * y = x^{\sigma} \cdot y^{\tau} \cdot c \quad .$$

Two correct conclusions in the Panny's cryptanalysis were given:

1. "the composition law in *any* entropic quasigroup comes from a multiplication in an abelian group that is twisted by automorphisms and translated by a constant."
2. "any non-associative power of an element $x \in G$ can in fact be written as a product combination in $(G, \cdot)$ of elements of the form x^{ψ} and c^{γ} where $\psi, \gamma \in \langle \sigma, \tau \rangle$."

The second conclusion was supported by the following Lemma:

Lemma 5. *For a binary operation $x * y = x^{\sigma} \cdot y^{\tau} \cdot c$ as in Theorem 1 and any non-associative exponent $\mathbf{A}$, there exists $\gamma \in \mathbb{Z}[\sigma, \tau]$ such that for all $x \in G$*

$$x^{\mathbf{A}} = x^{1+(\sigma+\tau-1)\gamma} \cdot c^{\gamma} \quad . \tag{4}$$

Moreover, if (4) holds for some $x = g \in G$, then (4) holds for all $x \in \langle g \rangle_*$.

Luckily (for the concept of Entropoid cryptography), Panny made one implicit assumption that the commutative automorphisms σ, τ of $(G, \cdot)$ are defined exclusively with the group operation of $(G, \cdot)$. That assumption led to the following two incomplete conclusions

1. "The classification of finite abelian groups implies that there exists a small subset of such elements that suffices to span the entire subquasigroup $\langle g \rangle_*$ generated by $g \in G$, and again, recovery of the exponents corresponding to Alice's private-key operation consists of a multidimensional discrete-logarithm computation (which is polynomial-time quantumly)."
2. "Therefore, all instantiations of the entropoid framework where a representation of $*$ using $\cdot$ and σ, τ, c can be found efficiently (cf. Section 2.2) should be breakable in polynomial time on a quantum computer."

With other words, assuming that automorphisms σ, τ of $(G, \cdot)$ are exclusively defined with the group operation $\cdot$ (basically taking that σ and τ are some fixed integer exponents), indeed the statements in Lemma 1 suggest that the secret power index $\mathbf{A}$ of Alice can be represented in a form that depends on another unknown fixed integer γ, and thus the nature of the problem remains the same: solving the Discrete Logarithm Problem.

However, Theorem 1 does not specify the nature of the automorphisms σ, τ of $(G, \cdot)$. For that matter, the group of all automorphisms $\mathbf{Aut}(G)$ can be very reach, and we can definitely find automorphisms that bijectively and homomorphically (regarding the group operation $\cdot$) are mapping the elements of G to G, but they can not be represented as fixed number of applications exclusively of the internal operation $\cdot$ of G. We used one such involutive automorphism in Definition 2.

In that case, the relation (4) still holds, but the recovery of the exponents corresponding to Alice's private-key operation becomes a search for a power index $\gamma = (\gamma_1, \gamma_2)$. That problem as we showed in previous section reduces to the problem of computing exponential congruences for which there is no polynomial-time quantum algorithm.

Moreover, *now we do not need to hide* the abelian group $(G, \cdot)$, nor the automorphisms σ and τ. Concretely, as in Lemma 3 we select two power indices $S = (s_1, s_2)$ and $U = (u_1, u_2)$ where $\mathrm{GCD}(s_1^2 - s_2^2, q) = 1$ and $\mathrm{GCD}(u_1^2 - u_2^2, q) = 1$ to define two commuting automorphisms $\sigma, \tau : G \mapsto G$. Then we can take some $c \in G$, and we can again define a non-commutative and non-associative operation

$$x * y = x^{\sigma} \cdot y^{\tau} \cdot c \quad .$$

Then we can apply the techniques from [4] to compute non-associative powers.

As a conclusion of this section, we want to give the following

Remark: Power indices defined in Definition 5 and the defined Arithmetic for power indices from Lemma 2 gives us convenience to work directly with power indices, avoiding much more expensive operations with entropic non-commutative and non-associative quasigroups. Additionally we can try to apply the **Entropic-Lift** recipe for many existing classical cryptographic schemes.

3 Attacks on the Exponential Congruences Problem (ECP)

In this section we adapt the known algorithms for solving ECP given in [19, 21].

Let us first highlight the differences with the ECP addressed in the open literature and the ECP in this work:

1. In [19, 21], the equation (2) is defined over finite field $\mathbb{F}_q$, where $q = p^k$, p a prime number and the number of elements in the multiplicative group $\mathbb{F}_q^*$ is $q - 1 = p^k - 1$. In our case the equation is over a ringoid structure called Entropoid, $\mathbb{E}_{q^2} = (G, \boxplus, \cdot)$ where $(G, \cdot)$ is a group with $q^2 = q_1^2 \dots q_\nu^2$ elements, $q_1, \dots, q_\nu$ prime numbers, and the operation $\boxplus$ is defined with an automorphism $T : G \mapsto G$ that can not be expressed as an exponentiation in G.

2. In [19, 21], g_1 and g_2 are in general different (but the authors also discuss the situations where $g_1 = g_2$) and have respectively orders s and t i.e., $|\langle g_1 \rangle| = s$ and $|\langle g_2 \rangle| = t$, and where $s, t < q$ but $st \geq q$, while in our case $g_1 = g_2 = g$ and $|\langle g \rangle| = q$.

3. Depending on chosen g_1 and g_2, in [19, 21] the equation can have zero, one or many solutions (x_1, x_2), while in our case for g generator of the Entropoid, there exists one unique solution (x_1, x_2).

Lemma 6 (adaptation of Theorem 1 in [21])**.** *Let g be a generator of $\mathbb{E}_{q^2}$. Further, let $y \in G$ be given such that it is a solution of the equation $y = g^{x_1} \boxplus g^{x_2}$, for some $(x_1, x_2) \in \mathbb{Z}_q \times \mathbb{Z}_q$. One can find the solution (x_1, x_2) in deterministic time $O(q^{3/2}(\log q))$.*

Proof. For every $x_2 \in \{0, 1, \dots, q-1\}$ we evaluate $y \boxminus g^{x_2}$ and then we try to compute its discrete logarithm to base g, that is an integer x_1 with $g^{x_1} = y \boxminus g^{x_2}$. A deterministic algorithm for this problem is the Shanks' Baby Step-Giant Step method [16] which runs in time $O(q^{1/2}(\log q))$ and space $O(q^{1/2})$. In our case the Baby Step-Giant Step method will give us either a solution, or will return that there is no solution for that particular x_2. Combining the run time to go trough all cases for x_2 and the time to solve each instance of the discrete logarithm problem gives us the total worst case complexity of finding the solution of the equation (3) which is $O(q\, q^{1/2}(\log q)) = O(q^{3/2}(\log q))$. $\square$

Corollary 2. *There is a randomized algorithm for finding a solution of the equation $y = g^{x_1} \boxplus g^{x_2}$ that takes $O(q_\nu^{5/2})$ group operations, where $q = q_1 \dots q_\nu$ and the largest prime factor is $q_\nu > q^{1/2}$.*

Proof. We replace the complexity $O(q^{1/2}(\log q))$ of the Baby Step-Giant Step method in Lemma 6, with a randomized algorithm for computing the discrete logarithm that takes $\Omega(\sqrt{q_\nu})$ group operations proposed in Shoup's work [18]. That makes the total running time for solving (2) to be $O(q\ q_\nu^{1/2}) < O(q_\nu^2\ q_\nu^{1/2}) = O(q_\nu^{5/2})$.

□

Lemma 7 (adaptation of Theorem 3 in [21])**.** *Let g be a generator of $\mathbb{E}_{q^2}$, and let $y \in G$ be given such that it is a solution of the equation $y = g^{x_1} \boxplus g^{x_2}$, for some $(x_1, x_2) \in \mathbb{Z}_q \times \mathbb{Z}_q$. One can find the solution (x_1, x_2) on a quantum computer in time $O(q^{1/2}(\log\log q))$.*

Proof. For every $x_2 \in \{0, 1, \ldots, q-1\}$ we evaluate $y \boxminus g^{x_2}$ and then we use Shor's algorithm [17] to compute its discrete logarithm to base g, that is an integer x_1 with $g^{x_1} = y \boxminus g^{x_2}$ or to report that no such x_1 exists. The expected number of attempts before Shor's algorithm gives us the answer is $O(\log\log q)$. Let denote by $\mathcal{S}(x_2)$ the subroutine that implements Shor's quantum circuit.

We now use Grover's search algorithm [6] over the subroutine $\mathcal{S}(x_2)$ and the search space $x_2 \in \{0, 1, \ldots, q-1\}$. The whole running time is then $O(q^{1/2}(\log\log q))$.

□

4 A suitable instance for a concrete Entropoid structure

For our concrete instantiation of the group $(G, \cdot)$ with $|G| = q^2$ elements we will use the multiplicative group $(G, \cdot) \equiv C(n, 2)$ of all non-singular $n \times n$ circulant matrices over $\mathbb{F}_2$. Circulant $n \times n$ matrices over any field form a ring with the operations matrix addition and matrix multiplication. That ring is isomorphic with the quotient ring of polynomials $R = \mathbb{F}_2[x]/(x^n - 1)$. For further reading about circulant matrices I suggest for example [7, 8] and the references there.

The following result is known about the number of elements of the group $C(n, 2)$.

Proposition 4 (see Corollary 13.2.34, p.505 of [10])**.** *Assume 2 and n are co-prime. Then*

$$|C(n,2)| = \prod_{j=1}^{r} (2^{m_j} - 1), \tag{5}$$

where $m_1, \ldots, m_r$ are the degrees of the irreducible factors of $x^n - 1$ over $\mathbb{F}_2$.

We are interested for cases where n is a prime number, where $|C(n, 2)| = q^2$, $q = 2^{\frac{n-1}{2}} - 1$, and when q represented as a product of ν prime factors $q = q_1 \ldots q_\nu$, (sorted in ascending order), the number of prime factors ν is as small as possible, but the biggest factor q_ν is a big prime number $q_\nu > q^{1/2}$.

In Table 2 we give 8 instances for different values of n. Note that the green highlighted values 479, 647 and 863 are proposed to be instances that offer at least the security of NIST's Level 1, 3 and 5. The instance with $n = 103$ defines q as a very smooth number, and it would be an easy challenge, while for challenges with $n = 167, 263$ and 359, I hope that the cryptology community will give a significant feedback and analysis.

n	q_1	ν	$\log_2(q_\nu)$	Note
103	103	7	16.9999	Small n with smooth q
167	167	2	75.6163	An n for a challenge
263	263	2	122.9611	An n for a challenge
359	359	3	160.0273	An n for a challenge
479	479	6	162.2824	An n for NIST Level 1
647	647	6	199.6601	An n for NIST Level 3
863	863	8	230.0670	An n for NIST Level 5
887	887	3	385.6449	Beyond NIST Level 5

Table 2: We omit the exact numerical values for all factors of q in order to present a compact table. For example for $n = 167$ we have that $q = q_1 * q_2$ where $q_1 = 167$ and $q_2 = 57912614113275649087721 \approx 2^{75.6163}$. The table shows some concrete instances for prime numbers n. The column q_1 contains the smallest prime factor of $q = 2^{\frac{n-1}{2}} - 1$ when the order of $C(n, 2)$ is computed by the expression (5). The column ν contains the number of prime factors of q i.e. $q = q_1 \ldots q_\nu$, and the column $\log_2(q_\nu)$ contains the size of the largest factor q_ν in bits.

For the involutive automorphism $T : C \mapsto C$ it turns out that the operation of matrix transposition of elements in C is a suitable operation. It is automorphism, and it is involution. When elements $a \in C$ are being presented as polynomials

$$a = a_0 + a_1x + a_2x^2 + \dots a_{n-2}x^{n-2} + a_{n-1}x^{n-1},$$

the transposition of a is denoted as a^T and

$$a^T = a_0 + a_{n-1}x + a_{n-2}x + \dots + a_2x^{n-2} + a_1x^{n-1}.$$

5 Examples of Entropic-Lift

Example 1. Entropic-Lift for classical Diffie-Hellman key exchange protocol is just a straightforward variables replacement. For the classical case, Alice and Bob agree on a finite cyclic group $(G, \cdot)$ of order n and a generating element $g \in G$. For the Entropic Diffie-Hellman, Alice and Bob agree on a finite Entropoid $\mathbb{E}_{q^2} = (G, \boxplus, \cdot)$ with q^2 elements and a generating element $g \in G$.

(a) Classical Diffie–Hellman key exchange	**(b)** Entropic Diffie–Hellman key exchange
1. Alice picks a random natural number a where $1 < a < n$, and sends the element g^a of G to Bob.	1. Alice picks a random power index $a = (a_1, a_2)$ where $1 < a_1, a_2 < q$, and sends the element g^a of G to Bob.
2. Bob picks a random natural number b where $1 < b < n$, and sends the element g^b of G to Alice.	2. Bob picks a random power index $b = (b_1, b_2)$ where $1 < b_1, b_2 < q$, and sends the element g^b of G to Alice.
3. Alice computes the element $SharedKey = \left(g^b\right)^a = g^{ba}$ of G.	3. Alice computes the element $SharedKey = \left(g^b\right)^a = g^{ba}$ of G.
4. Bob computes the element $SharedKey = (g^a)^b = g^{ab}$ of G.	4. Bob computes the element $SharedKey = (g^a)^b = g^{ab}$ of G.

Table 3: Classical and Entropic Diffie-Hellman key exchange protocol. The Entropic variant is just a straightforward variables replacement

Example 2. Entropic-Lift for Schnorr signature scheme [15] is also a straightforward variables replacement. In the classical case all users agree on a finite cyclic group $(G, \cdot)$ of prime order q and a generating element $g \in G$, in which the Discrete Logarithm Problem is assumed to be hard. Also, all users agree on a cryptographic hash function $H : \{0, 1\}^* \mapsto \mathbb{Z}_q$.

In the Entropic Schnorr case, all users agree on a finite Entropoid $\mathbb{E}_{q^2} = (G, \boxplus, \cdot)$ with q^2 elements and a generating element $g \in G$, in which the Exponential Congruences Problem is assumed to be hard. All users also agree on a cryptographic hash function $H : \{0, 1\}^* \mapsto \mathbb{Z}_q \times \mathbb{Z}_q$.

In Table 5 we present a variant of Entropic Schnorr signature scheme called SEQUOA that was implemented in C++ and submitted for inclusion in Supercop [1] for testing and measurement. Six different instances are submitted to Supercop with prime number $n = 167, 263, 359, 479, 647, 863$. The design goal for this variant was to offer some of the desirable features for signature schemes named as "BUFF - Beyond UnForgeability Features" [3].

Additional motivation was to offer a randomized signature scheme which in a case of complete deterioration of the entropy pool for its randomness, the scheme will become a deterministic scheme. That part is the step "Set $k = (k_1, k_2) \leftarrow H(rand||PrivateKey||M)$, where $rand \xleftarrow{\$} \{0, 1\}^n$ is a sequence of at least n randomly generated bits" in the signing part.

A more detailed description about the concrete implementation of SEQUOA will be given elsewhere.

(a) Classical Schnorr signature scheme	(b) Entropic Schnorr signature scheme
KeyGen	**KeyGen**
$PrivateKey \equiv x \xleftarrow{\$} \mathbb{Z}_q^*, \quad PublicKey \equiv y = g^x$	$PrivateKey \equiv x = (x_1, x_2) \xleftarrow{\$} \mathbb{Z}_q \times \mathbb{Z}_q,$ $PublicKey \equiv y = g^x$
Sign a message M	**Sign** a message M
Choose random $k \xleftarrow{\$} \mathbb{Z}_q^*$	Choose random $k \xleftarrow{\$} \mathbb{Z}_q \times \mathbb{Z}_q$
$r = g^k$	$r = g^k$
$e = H(r\|\|M)$	$e = H(r\|\|M)$
$s = k - xe$	$s = k - xe$
$Signature = (s, e)$	$Signature = (s, e)$
Verify	**Verify**
$r_v = g^s y^e$;	$r_v = g^s y^e$;
$e_v = H(r_v\|\|M)$	$e_v = H(r_v\|\|M)$
Return $True$ if $e_v = e$ else Return $False$	Return $True$ if $e_v = e$ else Return $False$

Table 4: Classical and Entropic Schnorr signature scheme. The Entropic variant is just a straightforward variables replacement.

KeyGen
$PrivateKey \equiv x = (x_1, x_2) \xleftarrow{\$} \mathbb{Z}_q \times \mathbb{Z}_q,$
$PublicKey \equiv y = g^x$

Sign a message M
Set $k = (k_1, k_2) \leftarrow H(rand||PrivateKey||M)$, where $rand \xleftarrow{\$} \{0,1\}^n$ is a sequence of at least n randomly generated bits.
$r = g^k$
$e = H(r||PublicKey||M)$
$s = k - xe$
$Signature = (s, e)$

Verify
$r_v = g^s y^e$;
$e_v = H(r_v||PublicKey||M)$
Return $True$ if $e_v = e$ else Return $False$

Table 5: SEQUOA variant of Entropic Schnorr signature scheme. This variant is BUFF friendly.

Example 3. Entropic-Lift for DSA scheme is not so straightforward. The definition of the scheme involves variables that in one part play a role in one group, and later they are interpreted as members of another group. That would cause incompatible arithmetic operations between power indices and group elements in the Entropoid structure.

If we face a situation of incompatible operation in the Entropoid case, as suggested in the Entropic-Lift recipe, we can try to replace the involved group element variable with a cryptographic hash of that variable that maps it to a power index, thus enabling a proper arithmetic operation between power indices. We should also re-evaluate the lifted scheme to check if that altered expression still makes sense and is secure.

We describe here the classical DSA, without much details of the bit sizes of some of the variables since they are not crucially important for the purpose of this example of Entropic-Lift transformation. For more details we redirect the user to see the detailed definition of DSA [14].

In the classical DSA case all users agree on an N-bit prime q and L-bit prime p such that $p - 1$ is multiple of q. A generator g is chosen to be in a form of $g = h^{(p-1)/q} \bmod p$. Also, all users agree on a cryptographic hash function $H : \{0,1\}^* \mapsto \mathbb{Z}_q$.

In the Entropic DSA case, all users agree on a finite Entropoid $\mathbb{E}_{q^2} = (G, \boxplus, \cdot)$ with q^2 elements and a generating element $g \in G$, in which the Exponential Congruences Problem is assumed to be hard. All users also agree on a cryptographic hash function $H : \{0,1\}^* \mapsto \mathbb{Z}_q \times \mathbb{Z}_q$.

Notice that in the Entropic variant the arithmetic expression for s does not have the term $x \cdot r$, but $x \cdot H(r)$. That is because in this case $x \in \mathbb{Z}_q \times \mathbb{Z}_q$, while $r \in G$, so the multiplication $x \cdot r$ is not well defined. On the other hand, $H(r) \in \mathbb{Z}_q \times \mathbb{Z}_q$, so the expression $x \cdot H(r)$ is a valid arithmetic operation.

(a) Classical DSA scheme	**(b)** Entropic DSA scheme
KeyGen	**KeyGen**
$PrivateKey \equiv x \xleftarrow{\$} \mathbb{Z}_q^*$,	$PrivateKey \equiv x = (x_1, x_2) \xleftarrow{\$} \mathbb{Z}_q \times \mathbb{Z}_q$,
$PublicKey \equiv y = g^x \bmod p$	$PublicKey \equiv y = g^x$
Sign a message M	**Sign** a message M
Choose random $k \xleftarrow{\$} \mathbb{Z}_q^*$	Choose random $k \xleftarrow{\$} \mathbb{Z}_q \times \mathbb{Z}_q$
$r = \left(g^k \bmod p\right) \bmod q$	$r = g^k$
$s = \left(k^{-1}\left(H(M) + x \cdot r\right)\right) \bmod q$	$s = k^{-1}\left(H(M) + x \cdot H(r)\right)$
$Signature = (r, s)$	$Signature = (r, s)$
Verify	**Verify**
$w = s^{-1} \bmod q$;	$v_1 = r^s$
$u_1 = H(M) \cdot w \bmod q$; $u_2 = r \cdot w \bmod q$	$v_2 = g^{H(M)} \cdot y^{H(r)}$
$v = (g^{u_1} y^{u_2} \bmod p) \bmod q$;	
Return $True$ if $v = r$ else Return $False$	Return $True$ if $v_1 = v_2$ else Return $False$

Table 6: Classical and Entropic DSA scheme.

Open Problem 1. *From the perspective of provable security, and specifically from the perspective of security of post-quantum cryptographic schemes, precisely formalize and analyze the potentials and limits of the Entropic-Lift transformation.*

6 Conclusions

We offered a construction of algebraic structures, where rising to the non-associative power indices is no longer tied with the Discrete Logarithm Problem, but with a problem that in the last two decades has been analyzed and does not have a quantum polynomial algorithm that solves it. The problem is called Exponential Congruences Problem.

We also developed Arithmetic for the power indices. As a result, we proposed a generic recipe guidelines that we named "Entropic-Lift" for transforming some of the existing classical cryptographic schemes that depend on the hardness of Discrete Logarithm Problem to post-quantum cryptographic schemes that will base their security on the hardness of the Exponential Congruences Problem.

We demonstrated the Entropic-Lift on three concrete examples: transforming the classical Diffie-Hellman key exchange, Schnorr and DSA signature schemes.

We also posted one open problem in relation to Entropic-Lift transformation recipe: to precisely formalize and analyze the potentials and limits of the transformation.

Acknowledgements

I would like to thank my former PhD student Mattia Veroni for long discussions that we spent during the Summer of 2022 analyzing the Panny's attack on the Entropoid schemes. I would also like to thank Daniel Nager for sharing his preprint [12] in the beginning of November 2022 and for his input that improved this text.

References

[1] Daniel J. Bernstein and Tanja Lange. (editors), eBACS: ECRYPT Benchmarking of Cryptographic Systems. accessed 3 March 2023. https://bench.cr.yp.to.

[2] Richard H Bruck. Some results in the theory of quasigroups. *Transactions of the American Mathematical Society*, 55:19–52, 1944.

[3] Cas Cremers, Samed Düzlü, Rune Fiedler, Marc Fischlin, and Christian Janson. Buffing signature schemes beyond unforgeability and the case of post-quantum signatures. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1696–1714. IEEE, 2021.

[4] Danilo Gligoroski. Entropoid Based Cryptography. 2021. https://eprint.iacr.org/2021/469.

[5] Danilo Gligoroski. Rebuttal to claims in section 2.1 of the eprint report 2021/583" entropoid-based cryptography is group exponentiation in disguise". *Cryptology ePrint Archive*, 2021.

[6] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.

[7] Irwin Kra and Santiago R Simanca. On circulant matrices. *Notices of the AMS*, 59(3):368–377, 2012.

[8] Ayan Mahalanobis. The discrete logarithm problem in the group of non-singular circulant matrices. 2010.

[9] Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone. *Handbook of applied cryptography*. CRC press, 2018.

[10] Gary L Mullen and Daniel Panario. *Handbook of finite fields*, volume 17. CRC press Boca Raton, 2013.

[11] DC Murdoch. Quasi-groups which satisfy certain generalized associative laws. *American Journal of Mathematics*, 61(2):509–522, 1939.

[12] Daniel Nager. On linearization attack of entropic quasigroups cryptography. Cryptology ePrint Archive, Paper 2022/1575, 2022. https://eprint.iacr.org/2022/1575.

[13] Lorenz Panny. Entropoids: Groups in disguise. 2021. https://eprint.iacr.org/2021/583.

[14] Shirley M Radack. Updated digital signature standard approved as federal information processing standard (fips) 186-3. 2009.

[15] Claus-Peter Schnorr. Efficient identification and signatures for smart cards. In *Advances in Cryptology—CRYPTO'89 Proceedings 9*, pages 239–252. Springer, 1990.

[16] Daniel Shanks. Class number, a theory of factorization, and genera. in 1969 number theory institute (proc. sympos. pure math., vol. xx, state univ. new york, stony brook, ny, 1969), 415–440. *Amer. Math. Soc., Providence, Rhode Island, USA*, 1971.

[17] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[18] Victor Shoup. Lower bounds for discrete logarithms and related problems. In *Advances in Cryptology—EUROCRYPT'97: International Conference on the Theory and Application of Cryptographic Techniques Konstanz, Germany, May 11–15, 1997 Proceedings 16*, pages 256–266. Springer, 1997.

[19] Igor E Shparlinski. On finding solutions to exponential congruences. *Bulletin of the Australian Mathematical Society*, 99(3):388–391, 2019.

[20] Kôshichi Toyoda. On Axioms of Linear Functions. *Proceedings of the Imperial Academy*, 17(7):221–227, 1941.

[21] Wim Van Dam and Igor E Shparlinski. Classical and quantum algorithms for exponential congruences. In *Theory of Quantum Computation, Communication, and Cryptography: Third Workshop, TQC 2008 Tokyo, Japan, January 30-February 1, 2008. Revised Selected Papers 3*, pages 1–10. Springer, 2008.