Non-Commutative Key Exchange Protocol

Luis Adrián Lizama-Pérez and J. Mauricio López R.

Abstract. We introduce a novel key exchange protocol based on non-commutative matrix multiplication defined in \( \mathbb{Z}_p^{n \times n} \). The security of our method does not rely on computational problems as integer factorization or discrete logarithm whose difficulty is conjectured. We claim that the unique eavesdropper’s opportunity to get the secret/private key is by means of an exhaustive search which is equivalent to the unsorted database search problem. Furthermore, we show that the secret/private keys become indistinguishable to the eavesdropper. Remarkably, to achieve a 512-bit security level, the keys (public/private) are of the same size when matrix multiplication is done over a reduced 8-bit size modulo. Also, we discuss how to achieve key certification and Perfect Forward Secrecy (PFS). Therefore, Lizama’s algorithm becomes a promising candidate to establish shared keys and secret communication between (IoT) devices in the quantum era.

Keywords: Non-commutative · matrix · cryptography

1 Introduction

In 2017 the National Institute of Standards and Technology (NIST) initiated a process to evaluate the cryptographic algorithms that will be used to support security in the quantum era. Unfortunately, most of the cryptosystems used today will become obsolete in the foreseeable future because they would be broken by quantum computers [1]. Shor’s algorithm [2] solves the mathematical problems on which cryptography is supported: integer factorization and discrete logarithm. Although quantum principles have threatened the security of major cryptographic systems, they have raised a new technology known as quantum key distribution (QKD) that allows remote secret key establishment [3,4,5,6].

Post-quantum crypto-systems under evaluation for public-key quantum-resistant [7] include cryptography based on lattices, multi-variate-based, hash-based [8,9] and code-based [10]. After the third evaluation round, NIST has selected seven algorithms (and eight alternative candidates), four of them are public key encryption (and key-establishment) systems and three correspond to digital signature algorithms. In the first category, CRYSTALS-KYBER, NTRU-HPS, SABER are lattice-based while Classic McEliece is a code-based public key encryption system. Regarding digital signature schemes, CRYSTALS-DILITHIUM and FALCON are lattice-based and Rainbow is a multivariate-based algorithm [11,12,13]. According to the criteria defined by NIST, quantum algorithms must be resistant against classical and quantum adversaries, their security level must be comparable to the security of SHA-385 and AES-256. Issues to be considered are the size of the keys and the required computing resources and facility of implementation (in hardware and software). Versatility of the algorithm will be evaluated because of its ability to encrypt messages, perform digital signatures and/or allow key exchange.
As discussed in [14], Lizama’s certification method is scalable and interoperable and can be exploited in the pre-quantum and quantum era because the protocol exhibits indistinguishability of the integers used in the public key and ciphertexts. Moreover, public keys size in Lizama’s protocol has the smallest size: 0.256 kilobytes and 0.384 kilobytes for public key and certified key, respectively [14].

In this work, we will introduce a new key exchange algorithm based in non-commutative matrix multiplication that can be useful for secret communication in the pre-quantum but also in the quantum era. The article is organized as follows: in Section 2 we discuss some related protocols starting with the Diffie-Hellman algorithm. In Section 3, we introduce our Non-Commutative Key Exchange Protocol (nc-KEP) to later introduce in Section 4 the generalized non-commutative KEP. Section 5 describes a process to certificate the public keys across interdomain certificates. Finally, Section 6 details our PFS method that guarantee secrecy of the new session keys.

2 Related protocols

Without wishing to discuss them exhaustively, in this section we will give a brief introduction to the main cryptographic key establishment methods. We will start from the Diffie-Hellman protocol, which we consider the starting point for subsequent protocols. We will briefly mention Quantum Key Distribution (QKD) and at the end of this section we describe a recently published commutative KEP that has driven the method introduced here.

2.1 Diffie-Hellman

Diffie-Hellman (DH) key exchange [15] works over a ring \( \mathbb{Z}_p \) with large order \( p \). The modulo \( p \) and the generator \( g \) which is primitive root in \( \mathbb{Z}_p \) are publicly shared. Alice chooses randomly an exponent integer \( x_a \) and computes \( k_a = g^{x_a} \mod p \) which she sends to Bob. Similarly, Bob obtains and responds to Alice with \( k_b = g^{x_b} \mod p \). Then each one of them performs exponentiation using the received number as incoming, such that Alice’s computes \( (g^{x_b} \mod p)^{x_a} \mod p = g^{x_a x_b} \mod p \) and Bob’s computes \( (g^{x_a} \mod p)^{x_b} \mod p = g^{x_a x_b} \mod p \) (see Figure 1). Both numbers are equal because modular exponentiation obeys the normal rules of ordinary exponentiation.

![Fig. 1: Diffie–Hellman protocol.](image_url)

The eavesdropper Eve would try to recover \( g^{ab} \) from \((g,G,g^a,g^b)\). One defines the Diffie-Hellman algorithm by \( F(g,G,g^a,g^b) = g^{ab} \). We say that a group \( G \) with large order \( p \) satisfies the Computational Diffie-Hellman (CDH) assumption if no efficient algorithm exists to compute \( F(g,G,g^a,g^b) = g^{ab} \) [16]. Close related to the Computational Diffie-Hellman (CDH) assumption is the Discrete Logarithm Problem (DLP) which is defined as recovering \( x \) given \( g \) and \( g^x \mod p \).
2.2 Stickel

Stickel’s key exchange protocol was motivated by the Diffie-Hellman protocol [15]. In the original formulation, the group used in the protocol was the group of invertible matrices over a finite field [17,18]. Let \( G \) be a public non-abelian finite group. Let \( a, b \in G \) be public elements such that \( ab \neq ba \). Let the orders of \( a \) and \( b \) be \( N \) and \( M \) respectively:

1. Alice chooses two random natural numbers \( n < N, m < M \) and sends \( u = a^n b^m \) to Bob.
2. Bob picks two random natural numbers \( r < N, s < M \) and sends \( v = a^r b^s \) to Alice.
3. Alice derives the key as \( K_A = a^n v b^m = a^{n+r} b^{m+s} \).
4. Bob computes \( K_B = a^r u b^s = a^{n+r} b^{m+s} \).

Unfortunately a linear algebra attack to this protocol has been published [19,18]. It is sufficient for the adversary to find matrices \( x \) and \( y \) such that \( xa = ax, yb = by \), and \( xu = y \), because \( x \) corresponds to \( a^{-n} \), while \( y \) equals \( b^m \) [20].

2.3 Anshel-Anshel-Goldfeld

It defines a cryptographic primitive that uses non-commutative subgroups of a given platform group with efficiently computable normal forms. It was implemented in the braid group. This scheme assumes that the Conjugacy Search Problem (CSP) is difficult enough, so it might be implemented in other groups [18]. Let \( G \) be a group and elements \( a_1, \ldots, a_m, b_1, \ldots, b_n \in G \) are public.

1. Alice picks a private \( u \in G \) as a word \( a = u(a_1, \ldots, a_m) \) in alphabet \( A^{\pm 1} \), encodes (by normal forms), and sends publicly \( b_1^a, \ldots, b_n^a \).
2. Bob takes a (secret) word \( b = v(b_1, \ldots, b_n) \) in alphabet \( B^{\pm 1} \), encodes (by normal forms), and sends publicly \( a_1^b, \ldots, a_m^b \).
3. To decode Alice computes \( a^b = u(a_1^b, \ldots, a_m^b) \) and Bob gets \( b^a = v(b_1^a, \ldots, b_n^a) \). The common secret key is \( a^{-1} a^b = a^{-1}(b^{-1} a b) = (a^{-1} b^{-1} a b) b = (b^a)^{-1} b \).

2.4 Jintai Ding

It uses the learning with errors (LWE/RLWE) problem to build a key exchange scheme considered post-quantum. The basic idea of the construction can be viewed as certain extension of Diffie-Hellman problem with errors [21] which does the same thing using the associativity and commutativity, namely,

\[
x^T M y = (x^T M y) = x^T (M y)
\]

where \( M \) is an \( n \times n \) matrix in \( \mathbb{Z}_q \) and \( x, y \) are vectors in \( \mathbb{Z}_q^n \). It is required to introduce small errors according to the LWE problem which is defined as follows: Let \( \mathbb{Z}_q \) denote the ring of integers modulo \( q \) and let \( \mathbb{Z}_q^n \) denote the set of \( n \)-vectors over \( \mathbb{Z}_q \). There is a certain unknown linear function \( f : \mathbb{Z}_q^n \rightarrow \mathbb{Z}_q \) such that when the input is a sample of pairs \( (x, y) \), where \( x \in \mathbb{Z}_q^n \) and \( y \in \mathbb{Z}_q \), we have with high probability \( y = f(x) \).

2.5 Bennett-Brassard (BB84)

Although quantum principles have threatened the security of major cryptographic systems [2], they have raised a new technology known as Quantum Key Distribution (QKD) that allows remote secret key establishment. QKD protocols exploit the principle that an eavesdropper cannot alter quantum communication without producing a detectable noise [3]. Post-processing methods have emerged to accelerate the rate of the secret bits [22,6].
2.6 Lizama’s ni-KEP

The public key of user $i$ has two components $(P_i, Q_i)$ where $P_i = p^{2x_i}k_i \mod n$ and $Q_i = q^{2y_i}k_i \mod n$. The value $x_i$ is chosen randomly while $y_i$ is computed according to the relation $y_i = \phi(n) - x_i + 1$ where $\phi(n)$ is the Euler’s function. The module $n$ is the product of three public integer primes, so that $n = pqr$ where $p$ and $q$ are small prime numbers and $r$ is a big integer prime [23,14]. The $x_i$ value constitutes along $k_i$, the private key of user $i$ where $k_i$ is an invertible integer in the ring $\mathbb{Z}_n$. The steps of the protocols are summarized as follows (see Figure 2):

1. Once public keys have been exchanged, the users perform two operations over the numbers received: exponentiation and multiplication as indicated in Table 1.

<table>
<thead>
<tr>
<th>User</th>
<th>Operation</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>$(p^{2x_i} \cdot k_i \mod n)^{x_i} \cdot (q^{2y_i} \cdot k_i \mod n)^{y_i} \equiv p^{2x_i+y_i}q^{2y_i+y_i} \cdot k_i \mod n$</td>
<td></td>
</tr>
<tr>
<td>Bob</td>
<td>$(p^{2x_i} \cdot k_i \mod n)^{x_i} \cdot (q^{2y_i} \cdot k_i \mod n)^{y_i} \equiv p^{2x_i+y_i}q^{2y_i+y_i} \cdot k_i \mod n$</td>
<td></td>
</tr>
</tbody>
</table>

2. To derive the right hand results of Table 1, Euler’s theorem is applied in $\mathbb{Z}_n$. The theorem is written in Eq.5 where $n = pqr$ and $\phi(n) = (p-1)(q-1)(r-1)$. Here, $k$ and $n$ are relative prime each other, so $k$ is an invertible integer in $\mathbb{Z}_n$. Thus, according to Equation 5 we have

$$k^{\phi(n)+1} = k^{\phi(n)} \cdot k^1 \equiv k.$$  \hspace{1cm} (1)

3. Users exchange the resulting value $p^{2x_iax_i}q^{2y_iby_i}k_i \mod n$, which is multiplied by the corresponding inverse $k_i^{-1}$ at each side to derive the secret shared key $p^{2x_iax_i}q^{2y_iby_i} \mod n$ as depicted in Figure 2.

3 Non-Commutative Key Exchange Protocol

Now, we proceed to introduce the non-commutative Key Exchange Protocol (nc-KEP) which is based on classical non-commutative matrix algebra defined in $\mathbb{Z}_p^{n\times n}$. The public key $P_i$ of user $i$ is computed as $P_i = k_iu^xk_i^{-1}$ where $k$ and $u$ are random invertible square matrices defined in $\mathbb{Z}_p^{n\times n}$. Matrix multiplication is performed using a publicly known prime modulo $p$. Exponentiation can be done since is known that $P = ku^{-1} \rightarrow Pu^x = ku^xk^{-1}$. The exponent $x_i$ is a random secret integer number, so the private key of a user $i$ is the pair $(x_i, k_i)$. The protocol behaves according the following steps:

1. Alice and Bob obtain a copy of their public keys from the web service. Then, they perform the operations indicated in Table 2. Exponentiation inside $k_iu^{x_i}k_i^{-1}$ to $x_j$ can be performed applying exponentiation by squaring as illustrated in Equation 2.

$$k_iu^{x_i}k_i^{-1} = k_iu^{x_i}k_i^{-1}$$

$$k_iu^{x_i}k_i^{-1} \ldots$$
Fig. 2: Lizama’s non-invertible KEP [23]. All operations are modulo \( n \). According to Euler’s theorem \( k^{\phi(n)+1} \mod n = k \) because \( k \) is an invertible integer in \( \mathbb{Z}_n \).

Furthermore, to compute the public key \( P_i = k_i^u x_i k_i^{-1} \) defined in \( \mathbb{Z}_n \times n \), is required to raise \( u \) to a big integer \( x_i \) (of at least 128 bits). For this purpose we choose that \( u \) be a diagonalizable matrix, therefore \( u = g d_u g^{-1} \), where \( d_u \) is the diagonal matrix, such that it holds Equation 3.

\[
    u^{x_i} = g d_u^{x_i} g^{-1} \quad (3)
\]

Table 2: These operations are performed by users defined in \( \mathbb{Z}_p^{n \times n} \).

<table>
<thead>
<tr>
<th>User</th>
<th>Operation</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>((k_a u^{x_a} k_a^{-1})^{x_b} = k_a u^{x_a} k_a^{-1})</td>
<td></td>
</tr>
<tr>
<td>Bob</td>
<td>((k_b u^{x_b} k_b^{-1})^{x_a} = k_b u^{x_a} k_b^{-1})</td>
<td></td>
</tr>
</tbody>
</table>

2. The resulting matrix \( k_i^u x_i k_i^{-1} \) is sent to the other user who applies the convenient multiplication (left and right hand sides) to get the shared key \( k_s = u^{x_i x_j} \) as depicted in Figure 3.

**Cryptosystem.** Encryption can be easily achieved as done by the Hill cipher system because the shared secret key \( k_s = u^{x_i x_j} \) can be properly inverted to decrypt a block message of size equal to the matrix \( k_s \), as written in Equation 4 defined in \( \mathbb{Z}_p^{n \times n} \). Since not every possible matrix is an invertible matrix, users must restart the protocol when they derive a non-invertible matrix. It is known that the Hill cipher is vulnerable to a known-plaintext attack, so we will demonstrate in Section 6 how to safely generate a new secret key from the current one.
3.1 Security Analysis

According to [18] the Conjugacy Search Problem (CSP) can be stated as: given a recursive presentation of a group $G$ and two conjugate elements $u, h \in G$, find out a particular element $k \in G$ such that $k^{-1}uk = h$. It also implies that there should be a way to disguise elements of $G$ so that it would be impossible to recover $k$ from $k^{-1}uk$ just by inspection. Furthermore, a derived problem of the Conjugacy Search Problem is the Decomposition Search Problem (DSP) that states: given two elements $w$ and $w'$ of a group $G$, find two elements $x$ and $y$ that would belong to a given subset (usually a subgroup) $A \subseteq G$ and satisfy $x \cdot w \cdot y = w'$, provided at least one such pair of elements exists. If we denote $kuk^{-1}$ by $uk$, it looks like the DLP [24].

In the nc-KEP, the public key is computed as $ku = h$ defined in $\mathbb{Z}_{p \times n}^n$, where the pair $(x, k)$ is the private key. Despite $h$ and $u$ are publicly known, in accordance with the conjugacy problem definition, the eavesdropper is forced to guess $k$ but also $v = u^x$ because $x$ is unknown. This is equivalent to say that neither $x$ nor $g$ are known in DLP which implies that secrecy does not rely in the size of the modulo $p$.

So, let us rewrite the conjugacy problem as: given a group $G$ defined in $\mathbb{Z}_{p \times n}^n$ and one conjugate element $h \in G$, find out $k, v, \in \mathbb{Z}_{p \times n}^n$ such that $k^{-1}vk = h$. As can be deduced, it involves more complexity than the conjugacy problem (or the decomposition search problem), so we argue that the eavesdropper is forced to obtain $(k, v)$ from $h$ in any other way than exhaustive search. This is equivalent to searching an unsorted database problem.

To demonstrate the computational complexity exhibited by the non-commutative KEP we provide in the Appendix of this document, an example around the simplest matrix case where $n = 2$. We found that in the general case each term of the public key depends on $n$ secret integers due to matrix multiplication.

$$P_{11} = (k_{11}g_{11} + \ldots k_{1n}g_{n1})(s_{11}t_{11} + \ldots s_{1n}t_{n1})d_{11}x \mod p$$

$$+ (k_{11}g_{1n} + \ldots k_{1n}g_{nn})(s_{11}t_{11} + \ldots s_{1n}t_{n1})d_{nn}^{-1} \mod p$$
Suppose a user acts as a malicious Eve, so after she establishes a key with the target user, say Alice, she obtains $\mathbf{u}^{x_a}$, however, Eve can compute $e = \mathbf{u}^{x_e}$ then she would try to obtain $x_a$ from $e^{x_a}$. If Eve gets $x_a$ she derives $\mathbf{v} = \mathbf{u}^{x_a}$, indeed she has:

\[
P_{n1} = (k_{n1}g_{1n} + \ldots k_{nn}g_{nn})(s_{11}t_{11} + \ldots s_{1n}t_{n1})d_{11}x \mod p \\
+ (k_{n1}g_{1n} + \ldots k_{nn}g_{nn})(s_{11}t_{11} + \ldots s_{nn}t_{nn})d_{nn}x \mod p
\]

\[
P_{1n} = (k_{11}g_{11} + \ldots k_{1n}g_{1n})(s_{11}t_{11} + \ldots s_{1n}t_{nn})d_{11}x \mod p \\
+ (k_{11}g_{11} + \ldots k_{1n}g_{nn})(s_{11}t_{11} + \ldots s_{nn}t_{nn})d_{nn}x \mod p
\]

\[
P_{nn} = (k_{11}g_{11} + \ldots k_{nn}g_{nn})(s_{11}t_{11} + \ldots s_{nn}t_{nn})d_{11}x \mod p \\
+ (k_{11}g_{11} + \ldots k_{nn}g_{nn})(s_{11}t_{11} + \ldots s_{nn}t_{nn})d_{nn}x \mod p
\]

This condition imposes a severe computational challenge to the eavesdropper: recovering $x$ and $n$ secret numbers given $P_{ij}$ and $d_{ij}$, that is beyond the Discrete Logarithm Problem: recovering $x$ given $g$ and $g^x \mod p$. Finally, we would like to say that in the next section we discuss other security capabilities of the algorithm, namely, indistinguishability of the secret key $k_a$ and the private key $(x, k)$.

### 3.2 Performance Analysis

The public key of user $i$ is computed multiplying $k_i$, $\mathbf{u}^{x_i}$ and $k_i^{-1}$ which are square matrices of size $n \times n$ defined in $\mathbb{Z}_p^n$. As a result, $|k_i| = |\mathbf{u}^{x_i}| = n^2 \mod p$ but we choose $|k_i| = 256$ to be resistent in the quantum era. The size of each matrix’s element is equal to $|p|$, which is the size of the modulo $p$ and the size $|p|$ is obtained from $\frac{|k_i^2|}{n^2}$. The secret shared key is derived from $\mathbf{u}^{x_i,x_j}$. Thus, we deduced that the security level is $|x_i| + |x_j|$, thus for the quantum era $|x_i| = |x_j| = 128$.

For example, if we want a security level of 256 bits and we choose $n = 4$, then $|p| = 16$ because $\frac{256}{16}$. Also, $|k_i| = |\mathbf{u}^{x_i}| = 256$ because $|k_i| = 16 \mod |p|$ where each matrix’s element takes 16 bits. The size of the public key is $|P| = 256$ and the private key occupies $|x_i| + |k_i| = 128 + 256 = 384$ bits. In this example, the computation of the key requires $x_i$ (or $x_j$) matrix multiplications over a 16-bit size modulo. The matrix $\mathbf{u}_i$ is a public diagonal matrix initialized with random integers in $\mathbb{Z}_p$ (more details will be given in the next section). Other parameter sizes are written in the Table 3.

Table 3: It is shown some parameter sizes when is chosen $|k_a| = 256$ as the security level for $n = 2, 4, 8$. Sizes are written in bits.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>$2 \times 2$</th>
<th>$4 \times 4$</th>
<th>$8 \times 8$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Public key</td>
<td>$</td>
<td>k_1\mathbf{u}^{x_1}k_1^{-1}</td>
<td>$</td>
</tr>
<tr>
<td>Private key $</td>
<td>x_i</td>
<td>+</td>
<td>k_i</td>
</tr>
<tr>
<td>$</td>
<td>p</td>
<td>$</td>
<td>64</td>
</tr>
</tbody>
</table>

### 4 Generalized non-commutative KEA

Suppose a user acts as a malicious Eve, so after she establishes a key with the target user, say Alice, she obtains $\mathbf{u}^{x_a}$, however, Eve can compute $e = \mathbf{u}^{x_e}$ then she would try to obtain $x_a$ from $e^{x_a}$. If Eve gets $x_a$ she derives $\mathbf{v} = \mathbf{u}^{x_a}$, indeed she has:
— From the public channel: \( h_0 = k_x v^x k_0^{-1} \)
— From Alice’s public key: \( h_a = k_y v k_a^{-1} \)

Since \( v, v^x, h_0 \) and \( h_a \) are known matrices, the opponent can solve the system for \( k_x \) which is the Alice’s private key thus impersonating her. We would suggest that the size of \( x_a \) must be increased to 256 bits, but this attack has changed the unsorted database problem to a hardest version of the discrete logarithm problem based on matrices \([25,16]\). To avoid this attack we will introduce a generalized non-commutative KEP. Here, each user version of the discrete logarithm problem based on matrices \([25,16]\). To avoid this attack we increased to 256 bits, but this attack has changed the unsorted database problem to a hardest version of the discrete logarithm problem based on matrices \([25,16]\). To avoid this attack we increased to 256 bits, but this attack has changed the unsorted database problem to a hardest version of the discrete logarithm problem based on matrices \([25,16]\).

### 4.1 Indistinguishably of the secret key

Now, we want to demonstrate that the pair \((x, y)\) in the secret key \( k_x \) is indistinguishable from other pairs, symbolically \( e_1 x_1, e_2 y_1 = e_1 x_2, e_2 y_2 \), then for \( t = 1, 2 \) we can write:

\[- e_1 x_1 = u^{x_1} x_1 \]
\[- e_2 y_1 = w^{y_1} y_1 \]

From \( e_1 x_1, e_2 y_1 = e_1 x_2, e_2 y_2 \) we can rewrite it as \( u^{x_1} w^{y_1} = u^{x_2} w^{y_2} \) where \( x_t = x_{e_1} x_{a_1} \) and \( y_t = y_{e_1} y_{a_1} \) for \( t = 1, 2 \). In order to be indistinguishable, we must establish \( x_1 \neq x_2 \) and \( y_1 \neq y_2 \). But \( u \) and \( w \) are diagonalizable matrices, thus we can separate each equation’s term into factors. If we take the first term of the left hand side, we can factorize it as \( u^{x_1} = s_1 s_2 \) then:

\[- s_1 = g d_u x_1^1 \lambda g^{-1} \]
\[- s_2 = g d_u y_1^1 \lambda g^{-1} \]

Because \( x_1 - \lambda + \lambda = x_1 \) for \( \lambda = 0 \ldots x_1 \). Provided \( |x_1| = 256 \), we can separate into several factors each equation’s term. By separating them, we directly find \((x_1, y_1)\) and \((x_2, y_2)\), therefore the numbers \((x_1, y_1)\) in the secret key, become indistinguishable.

### 4.2 Indistinguishingly of the private key

A public key \((P, Q)\) is computed using the private key \((x, y, k)\). Indeed \((x, k)\) produces \( P \) while \((y, k)\) generates \( Q \). Suppose we have found another pair \((x, k)\) that also generates \( P \). We would like to show that \( y \) exists such that \((y, k)\) produces \( Q \). In other words, the private key become indistinguishable from the opponent’s point of view.

The public key is computed as \( P = k_x g d_u x_1 g^{-1} k_1^{-1} \) and \( Q = k_y h d_w y_1 h^{-1} k_2^{-1} \). But provided \((x, k)\) produces \( P \), we must find \( y \) such that \( P = k_y h d_w y_1 h^{-1} k_2^{-1} \). Therefore, we require that \( h d_w y_1^1 \neq h d_w y_1 \) for \( i \neq j \). Removing \( h \) both sides we have \( d_w y_1^1 \neq d_w y_1 \) which implies that each diagonal element satisfies the condition \( d_w y_1^1 \neq d_w y_1 \mod p \) for \( i = 1 \ldots n \) where \( n \) is the matrix dimension. To surpass such requirement each diagonal element will be computed as \( 2^{m} \) so that \( 2^{m} < p \) for \( i = 1 \ldots m \). Thus, each diagonal element of \( d_w \) is a power of the primitive root inside \( \mathbb{Z}_p \).

In this enhanced scenario, the size of the public key yields \( |(P, Q)| = 512 \), the private key \( |(x, y, k)| = 512 \) and the secret key raises its security level from 256 to 512 bits. Just to have...
a reference, Lizama’s ni-KEP [14] defines a public key size of 2048 bits and a private key size of 1280 bits which have the smallest when is compared against NIST Round 3 finalists [13].

As it can be concluded from this discussion, the generalized nc-KEP can be directly upgraded from its previous particular case. In the next sections we will use the non-generalized nc-KEP, so that a better explanation could be provided.

Table 4: The public keys in the generalized nc-KEP. The secret key between users will be $k_s = u^{x_a x_b w y_a y_b}$ defined in $\mathbb{Z}_p^{n \times n}$.

<table>
<thead>
<tr>
<th>User</th>
<th>P_i</th>
<th>Q_i</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>$k_a u^{x_a k_a^{-1}}$</td>
<td>$k_a w^{y_a k_a^{-1}}$</td>
</tr>
<tr>
<td>Bob</td>
<td>$k_b u^{x_b k_b^{-1}}$</td>
<td>$k_b w^{y_b k_b^{-1}}$</td>
</tr>
</tbody>
</table>

5 Certificated Keys

An indispensable property of public keys is to be authenticated by a Certification Authority (CA). The keys of the non-commutative Key Exchange Protocol (nc-KEP) can be certified if a CA raises the keys to her private key number $x_{ca}$ as indicated in Table 5. Alice and Bob obtain a copy of their public certificated keys from the web service. Then, they perform the usual exponentiation $(u^{x_{ca}})^{x_i}$. The secret shared key is derived as $u^{x_{ca} x_{ca} x_j}$.

Table 5: CA’s public database. CA performs exponentiation over the public keys. The secret shared key is $k_s = u^{x_{ca} x_{ca} x_j}$ defined in $\mathbb{Z}_p^{n \times n}$.

<table>
<thead>
<tr>
<th>User</th>
<th>Public key</th>
<th>Certified key</th>
</tr>
</thead>
<tbody>
<tr>
<td>CA</td>
<td>$k_{ca} u^{x_{ca} k_{ca}^{-1}}$</td>
<td>-</td>
</tr>
<tr>
<td>Alice</td>
<td>$k_a u^{x_a k_a^{-1}}$</td>
<td>$k_a u^{x_{ca} x_{ca} k_a^{-1}}$</td>
</tr>
<tr>
<td>Bob</td>
<td>$k_b u^{x_b k_b^{-1}}$</td>
<td>$k_b u^{x_{ca} x_{ca} k_b^{-1}}$</td>
</tr>
</tbody>
</table>

5.1 Interdomain certificates

Users that have been certified with different Certification Authorities, say CA$_1$ and CA$_2$ can establish a secret key, if each CA certifies their keys with the converse CA. It means that after the second certification, the public key of users can be written as $k_i u^{x_{ca_1} x_{ca_2}} k_i^{-1}$ and $k_j u^{x_{ca_1} x_{ca_2}} k_j^{-1}$ of user $i$ and $j$, respectively. The shared secret key between users will be $u^{x_{ca_1} x_{ca_2} x_j}$ defined in $\mathbb{Z}_p^{n \times n}$. 

6 Perfect Forward Secrecy (PFS)

Remote users Alice and Bob may want to establish a new secret key $k_t$ based on the they already have $k_s$. However, if for some reason $k_t$ is compromised by an opponent, Perfect Forward Secrecy (PFS) is a property of key agreement protocols that guarantee that such leakage does not compromise the security of previously used keys. In Figure 4 we depict our PFS protocol to produce new session secret keys. Provided the private keys $k_a$ and $k_b$ remain secret, the eavesdropper could capture $k_t$ but she does not know $t_a, t_b$ thus she cannot derive $k_s$.

![Diagram of PFS protocol](image)

**Fig. 4:** Perfect Forward Secrecy (PFS) in Lizama’s non-commutative Key Exchange Protocol (nc-KEP). The new shared secret key between users is $k_t = k_s t_a t_b$ defined in $\mathbb{Z}_{np}^{n \times n}$. Symbols in the figure denote matrices.

7 Conclusions

We introduced here the non-commutative key exchange protocol (nc-KEP) which allows the secret key establishment between two remote parties in order to enable private communication. Lizama’s nc-KEP does not rely on computational problems as integer factorization or discrete logarithm whose complexity is conjectured. By contrast, we have evaluated the security of the method taking as a reference the Conjugacy Search Problem. We have showed the computational complexity that arises the involved matrix multiplication.

The generalized nc-KEP achieves 512-bit security level when the public and private keys reach the same size while matrix multiplications are done over a reduced 8-bit size modulo. Moreover, we have demonstrated our method exhibits Perfect Forward Secrecy (PFS).

Therefore, Lizama’s nc-KEP enables secret communication between restricted computational IoT devices in the quantum era. The algorithm would be further optimized in hardware/software since it basically requires matrix-multiplication.

8 Appendix

To better illustrate the computational complexity required to cryptanalyze the nc-KEP, let us consider the minimum matrices allowed by the protocol, that is $2 \times 2$. A user’s public key $P_i$ is
Non-Commutative Key Exchange Protocol 11

represented as $P_1 = k_u u^x_1 k_1^{-1}$ where $u$ is a diagonalizable matrix that can be written as $g d_u g^{-1}$, therefore $u^x = g d_u^x g^{-1}$, where $d_u$ is the diagonal matrix. Then we have $P_1 = k_i g d_u x^i g^{-1} k_i^{-1}$ in $Z_p^{n \times n}$. Now we proceed according to Equation 5.

\[
P_1 = \begin{bmatrix}
  k_{11} & k_{12} \\
  k_{21} & k_{22}
\end{bmatrix}
\begin{bmatrix}
  g_{11} & g_{12} \\
  g_{21} & g_{22}
\end{bmatrix}
\begin{bmatrix}
  d_1^{x_i} & d_2^{x_i} \\
  d_2^{x_i} & d_1^{x_i}
\end{bmatrix}
\begin{bmatrix}
  g_{11} & g_{12} \\
  g_{21} & g_{22}
\end{bmatrix}^{-1}
\begin{bmatrix}
  k_{11} & k_{12} \\
  k_{21} & k_{22}
\end{bmatrix}^{-1} \mod p
\]

(5)

\[
P_i = \begin{bmatrix}
  k_{11} & k_{12} \\
  k_{21} & k_{22}
\end{bmatrix}
\begin{bmatrix}
  g_{11} & g_{12} \\
  g_{21} & g_{22}
\end{bmatrix}
\begin{bmatrix}
  d_1^{x_i} & d_2^{x_i} \\
  d_2^{x_i} & d_1^{x_i}
\end{bmatrix}
\begin{bmatrix}
  s_1 & s_2 \\
  s_2 & s_1
\end{bmatrix}
\begin{bmatrix}
  t_1 & t_2 \\
  t_2 & t_1
\end{bmatrix} \mod p
\]

where $s = g^{-1}$ and $t = k^{-1}$. Thus, we deduce the following relations for the elements of $P_i$:

\[
P_{i1} = (s_1 t_1 + s_2 t_2)(k_{11} g_{11} + k_{12} g_{21}) d_1^{x_i} \mod p + (s_2 t_1 + s_2 t_2)(k_{11} g_{12} + k_{12} g_{22}) d_2^{x_i} \mod p
\]

\[
P_{i2} = (s_1 t_1 + s_2 t_2)(k_{11} g_{11} + k_{12} g_{21}) d_1^{x_i} \mod p + (s_2 t_1 + s_2 t_2)(k_{11} g_{12} + k_{12} g_{22}) d_2^{x_i} \mod p
\]

\[
P_{21} = (s_1 t_1 + s_2 t_2)(k_{11} g_{11} + k_{12} g_{21}) d_1^{x_i} \mod p + (s_2 t_1 + s_2 t_2)(k_{21} g_{12} + k_{22} g_{22}) d_2^{x_i} \mod p
\]

\[
P_{22} = (s_1 t_1 + s_2 t_2)(k_{11} g_{11} + k_{12} g_{21}) d_1^{x_i} \mod p + (s_2 t_1 + s_2 t_2)(k_{21} g_{12} + k_{22} g_{22}) d_2^{x_i} \mod p
\]

As can be seen there, each term of the public key depends on a number of secret integers due to matrix multiplication. For example, $P_{i11}$ depends on $k_{11}$, $k_{12}$, $t_1$, and $t_2$ which are kept secret. Provided the elements of $k_1$ remain private, so the elements of $k_1^{-1}$.

We do not devise a method to derive the private key $(x_i, k_i)$ from $P_i$ unless it is done by exhaustive search which is equivalent to the unsorted database search problem that is at least exponential-time in the length of $k$ and $x$, thus infeasible for practical purposes. As long as the total size of $k$ is at least 256 bits, the method is assumed to be resistant to the Grover’s quantum search [26].

References