 doi:10.20944/preprints202105.0174.v1

Non-Commutative Key Exchange Protocol

Luis Adrián Lizama-Pérez[1][0000−0001−5109−2927] and J. Mauricio López R.[2]

Sección de Posgrado de la Universidad Politécnica de Pachuca,
Ex-Hacienda de Santa Bárbara, 43830, México
luislizama@upp.edu.mx
Cinvestav Querétaro, Libramiento Norponiente 2000,
Real de Juriquilla, 76230, Santiago de Querétaro, Querétaro, México
jm.lopez@cinvestav.mx

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 $k_a = g^{x_a} \bmod p$ which she sends to Bob. Similarly, Bob obtains and responds to Alice with $k_b = g^{x_b} \bmod p$. Then each one of them performs exponentiation using the received number as incoming, such that Alice's computes $(g^{x_b} \bmod p)^{x_a} \bmod p = g^{x_b x_a} \bmod p$ and Bob's computes $(g^{x_a} \bmod p)^{x_b} \bmod p = g^{x_a x_b} \bmod p$ (see Figure 1). Both numbers are equal because modular exponentiation obeys the normal rules of ordinary exponentiation.

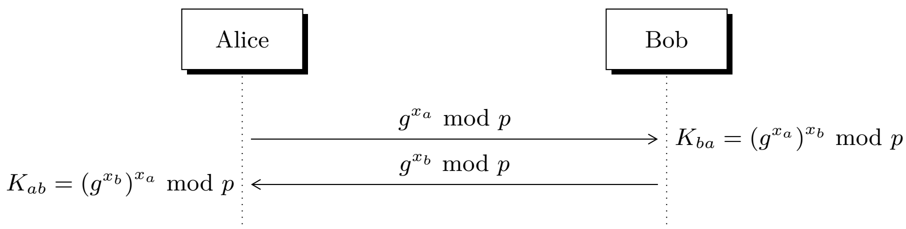

Fig. 1: Diffie–Hellman protocol.

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 $g^x \bmod p$.

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 $a^{-1}a^b = a^{-1}(b^{-1}ab) = (a^{-1}b^{-1}a)b = (b^a)^{-1}b$.

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,

$$\mathbf{x}^T\mathbf{M}\mathbf{y} = (\mathbf{x}^T\mathbf{M})\mathbf{y} = \mathbf{x}^T(\mathbf{M}\mathbf{y})$$

where $\mathbf{M}$ is an $n \times n$ matrix in $\mathbb{Z}_q$ and $\mathbf{x}$, $\mathbf{y}$ 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 $f : \mathbb{Z}_q^n \to \mathbb{Z}_q$ such that when the input is a sample of pairs $(\mathbf{x}, y)$, where $\mathbf{x} \in \mathbb{Z}_q^n$ and $y \in \mathbb{Z}_q$, we have with high probability $y = f(\mathbf{x})$.

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 $P_i = p^{2x_i} k_i \bmod n$ and $Q_i = q^{y_i} k_i \bmod n$. 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 $n = p\ q\ r$ 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 1: Exponentiation and multiplication are performed by users after their public keys have been exchanged.

User	Operation	Result
Alice	$\left(p^{2x_b} \cdot k_b \bmod n\right)^{x_a} \cdot (q^{y_b} \cdot k_b \bmod n)^{y_a} \equiv$	$p^{2x_b x_a} q^{y_b y_a} \cdot k_b \bmod n$
Bob	$\left(p^{2x_a} \cdot k_a \bmod n\right)^{x_b} \cdot (q^{y_a} \cdot k_a \bmod n)^{y_b} \equiv$	$p^{2x_a x_b} q^{y_a y_b} \cdot k_a \bmod n$

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 $n = p\ q\ r$ 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 $k^{\phi(n)+1} = k^{\phi(n)} \cdot k^1 = k$.

$$k^{\phi(n)} \equiv 1 \bmod n \tag{1}$$

3. Users exchange the resulting value $p^{2x_a x_b} q^{y_a y_b} k_i \bmod n$, which is multiplied by the corresponding inverse ${k_i}^{-1}$ at each side to derive the secret shared key $p^{2x_a x_b} q^{y_a y_b} \bmod n$ 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 $\mathbf{P_i}$ of user i is computed as $\mathbf{P_i} = \mathbf{k_i} \mathbf{u}^{x_i} \mathbf{k_i}^{-1}$ where $\mathbf{k}$ and $\mathbf{u}$ 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 $\mathbf{P} = \mathbf{k}\mathbf{u}\mathbf{k}^{-1} \rightarrow \mathbf{P}^x = \mathbf{k}\mathbf{u}^x\mathbf{k}^{-1}$. The exponent x_i is a random secret integer number, so the private key of a user i is the pair $(x_i, \mathbf{k_i})$. 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 $\mathbf{k_i}\mathbf{u}^{x_i}\mathbf{k_i}^{-1}$ to x_j can be performed applying exponentiation by squaring as illustrated in Equation 2.

$$\begin{aligned} \mathbf{k_i}\mathbf{u}^{x_i\, x_j}\mathbf{k_i}^{-1} = \mathbf{k_i}\mathbf{u}^{x_i}\mathbf{k_i}^{-1} & \\ & \mathbf{k_i}\mathbf{u}^{x_i}\mathbf{k_i}^{-1} \ldots \end{aligned} \tag{2}$$

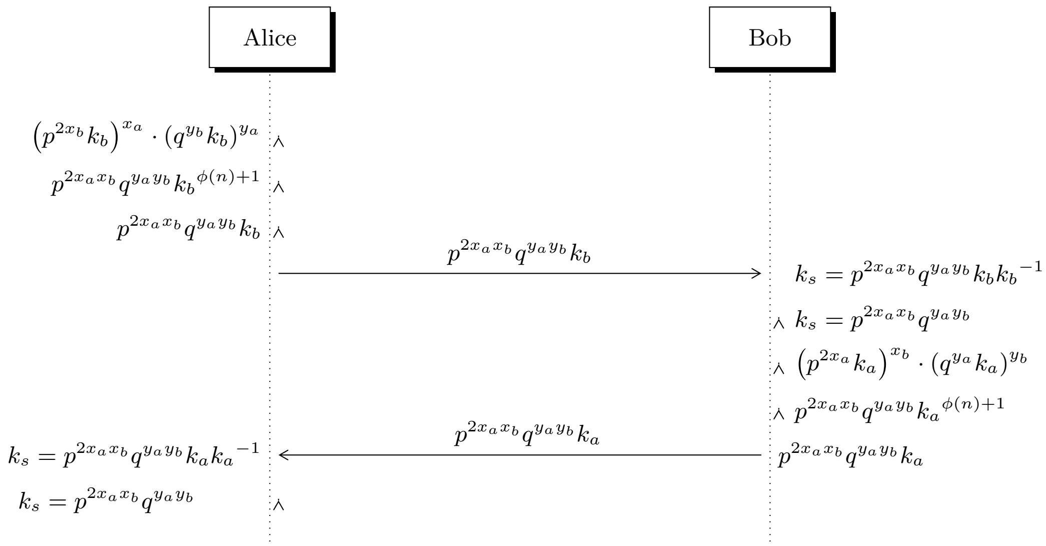

Fig. 2: Lizama's non-invertible KEP [23]. All operations are modulo n. According to Euler's theorem $k^{\phi(n)+1} \bmod n = k$ because k is an invertible integer in $\mathbb{Z}_n$.

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

$$\mathbf{u}^{x_i} = \mathbf{g}{\mathbf{d_u}}^{x_i}\mathbf{g}^{-1} \tag{3}$$

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

User	Operation	Result
Alice	$\left(\mathbf{k_b}\mathbf{u}^{x_b}\mathbf{k_b}^{-1}\right)^{x_a} =$	$\mathbf{k_b}\mathbf{u}^{x_ax_b}\mathbf{k_b}^{-1}$
Bob	$\left(\mathbf{k_a}\mathbf{u}^{x_a}\mathbf{k_a}^{-1}\right)^{x_b} =$	$\mathbf{k_a}\mathbf{u}^{x_ax_b}\mathbf{k_a}^{-1}$

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

Cryptosystem. Encryption can be easily achieved as done by the Hill cipher system because the shared secret key $\mathbf{k_s} = \mathbf{u}^{x_ax_b}$ can be properly inverted to decrypt a block message of size equal to the matrix $\mathbf{k_s}$, 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.

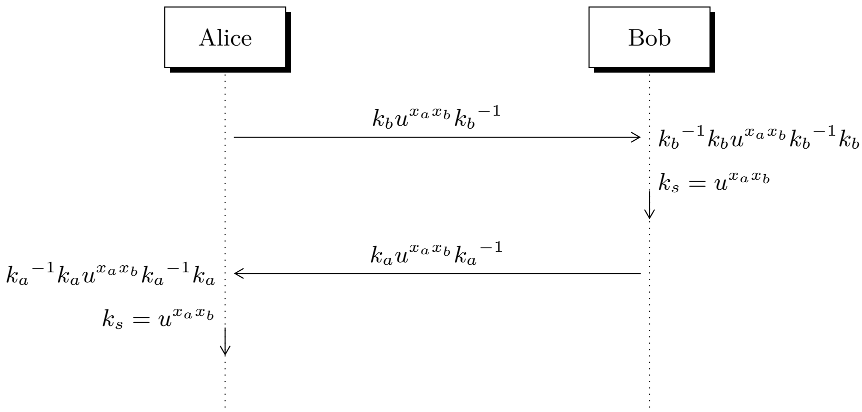

Fig. 3: Lizama's non-commutative Key Exchange Protocol (nc-KEP). The shared secret key between remote users is $\mathbf{k_s} = \mathbf{u}^{x_a x_b}$. Symbols in the figure denote matrices.

$$\begin{aligned} \mathbf{c} &= \mathbf{m}\mathbf{k_s} \\ \mathbf{m} &= \mathbf{c}\mathbf{k_s}{}^{-1} \end{aligned} \tag{4}$$

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 u^k, it looks like the DLP [24].

In the nc-KEP, the public key is computed as $\mathbf{k}\mathbf{u}^x\mathbf{k}^{-1} = \mathbf{h}$ defined in $\mathbb{Z}_p^{n\times n}$, where the pair $(x, \mathbf{k})$ is the private key. Despite $\mathbf{h}$ and $\mathbf{u}$ are publicly known, in accordance with the conjugacy problem definition, the eavesdropper is forced to guess $\mathbf{k}$ but also $\mathbf{v} = \mathbf{u}^x$ 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 $\mathbb{Z}_p^{n\times n}$ and one conjugate element $\mathbf{h} \in G$, find out $\mathbf{k}, \mathbf{v}, \in \mathbb{Z}_p^{n\times n}$ such that $\mathbf{k}^{-1}\mathbf{v}\mathbf{k} = \mathbf{h}$. 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 $(\mathbf{k}, \mathbf{v})$ from $\mathbf{h}$ 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.

$$\begin{aligned} &P_{11} = (k_{11}g_{11} + \ldots k_{1n}g_{n1})(s_{11}t_{11} + \ldots s_{1n}t_{n1})d_{11}{}^x \bmod p \\ &+ (k_{11}g_{1n} + \ldots k_{1n}g_{nn})(s_{n1}t_{11} + \ldots s_{nn}t_{n1})d_{nn}{}^x \bmod p \end{aligned}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 May 2021 doi:10.20944/preprints202105.0174.v1

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

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

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

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 $g^x \bmod p$. 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 $\mathbf{k_s}$ and the private key $(x, \mathbf{k})$.

3.2 Performance Analysis

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

For example, if we want a security level of 256 bits and we choose $n = 4$, then $\mid p \mid = 16$ because $\frac{256}{16}$. Also, $\mid \mathbf{k_i} \mid = \mid \mathbf{u_i}^{x_i} \mid = 256$ because $\mid \mathbf{k_i} \mid = 16 \mid p \mid$ where each matrix's element takes 16 bits. The size of the public key is $\mid \mathbf{P_i} \mid = 256$ and the private key occupies $\mid x_i \mid + \mid \mathbf{k_i} \mid = 128+256 = 384$ 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 $\mid \mathbf{k_s} \mid = 256$ as the security level for $n = 2, 4, 8$. Sizes are written in bits.

Parameter	2×2	4×4	8×8
Public key $\mid \mathbf{k_i u}^{x_i} {\mathbf{k_i}}^{-1} \mid$	256	256	256
Private key $\mid x_i \mid + \mid \mathbf{k_i} \mid$	384	384	384
$\mid p \mid$	64	16	4

4 Generalized non-commutative KEP

Suppose a user acts as a malicious Eve, so after she establishes a key with the target user, say Alice, she obtains $\mathbf{u}^{x_a x_e}$. However, Eve can compute $\mathbf{e} = \mathbf{u}^{x_e}$ then she would try to obtain x_a from $\mathbf{e}^{x_a}$. If Eve gets x_a she derives $\mathbf{v} = \mathbf{u}^{x_a}$, indeed she has:

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 10 May 2021 doi:10.20944/preprints202105.0174.v1

— From the public channel: $\mathbf{h_e} = \mathbf{k_e}\mathbf{v}^{x_e}\mathbf{k_e}^{-1}$
— From Alice's public key: $\mathbf{h_a} = \mathbf{k_a}\mathbf{v}\mathbf{k_a}^{-1}$

Since $\mathbf{v}$, $\mathbf{v}^{x_e}$, $\mathbf{h_e}$ and $\mathbf{h_a}$ are known matrices, the opponent can solve the system for $\mathbf{k_a}$ 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 i has two public matrices $(\mathbf{P_i}, \mathbf{Q_i})$ as they are shown in Table 4. Thus, the secret key between i and j is deduced to be $\mathbf{k_s} = \mathbf{u}^{x_i x_j}\mathbf{w}^{y_i y_j}$ scaling the complexity problem to the generalized relation $\mathbf{e_1}^{x_a}\mathbf{e_2}^{y_a}$ where $\mathbf{e_1} = \mathbf{u}^{x_e}$ and $\mathbf{e_2} = \mathbf{w}^{y_e}$.

4.1 Indistinguishably of the secret key

Now, we want to demonstrate that the pair (x_a, y_a) in the secret key $\mathbf{k_s}$ is indistinguishable from other pairs, symbolically $\mathbf{e_1}^{x_{a_1}}\mathbf{e_2}^{y_{a_1}} = \mathbf{e_1}^{x_{a_2}}\mathbf{e_2}^{y_{a_2}}$, then for $t = 1, 2$ we can write:

— $\mathbf{e_1}^{x_{a_t}} = \mathbf{u}^{x_{e_t} x_{a_t}}$
— $\mathbf{e_2}^{y_{a_t}} = \mathbf{w}^{y_{e_t} y_{a_t}}$

From $\mathbf{e_1}^{x_{a_1}}\mathbf{e_2}^{y_{a_1}} = \mathbf{e_1}^{x_{a_2}}\mathbf{e_2}^{y_{a_2}}$ we can rewrite it as $\mathbf{u}^{x_1}\mathbf{w}^{y_1} = \mathbf{u}^{x_2}\mathbf{w}^{y_2}$ where $x_t = x_{e_t} x_{a_t}$ and $y_t = y_{e_t} y_{a_t}$ for $t = 1, 2$. In order to be indistinguishable, we must establish $x_1 \neq x_2$ and $y_1 \neq y_2$. But $\mathbf{u}$ and $\mathbf{w}$ 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 $\mathbf{u}^{x_1} = \mathbf{s_1}\mathbf{s_2}$ then:

— $\mathbf{s_1} = \mathbf{g}\mathbf{d_u}^{x_1 - \lambda}\mathbf{g}^{-1}$
— $\mathbf{s_2} = \mathbf{g}\mathbf{d_u}^{\lambda}\mathbf{g}^{-1}$

Because $x_1 - \lambda + \lambda = x_1$ for $\lambda = 0 \dots x_1$. 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_i, y_i) in the secret key, become indistinguishable.

4.2 Indistinguishably of the private key

A public key $(\mathbf{P_{a_1}}, \mathbf{Q_{a_1}})$ is computed using the private key $(x_{a_1}, y_{a_1}, \mathbf{k_{a_1}})$. Indeed $(x_{a_1}, \mathbf{k_{a_1}})$ produces $\mathbf{P_{a_1}}$ while $(y_{a_1}, \mathbf{k_{a_1}})$ generates $\mathbf{Q_{a_1}}$. Suppose we have found another pair $(x_{a_2}, \mathbf{k_{a_2}})$ that also generates $\mathbf{P_{a_1}}$. We would like to show that y_{a_2} exists such that $(y_{a_2}, \mathbf{k_{a_2}})$ produces $\mathbf{Q_{a_1}}$. In other words, the private key become indistinguishable from the opponent's point of view.

The public key is computed as $\mathbf{P_{a_1}} = \mathbf{k_{a_1}}\mathbf{g}\mathbf{d_u}^{x_{a_1}}\mathbf{g}^{-1}\mathbf{k_{a_1}}^{-1}$, $\mathbf{Q_{a_1}} = \mathbf{k_{a_1}}\mathbf{h}\mathbf{d_w}^{y_{a_1}}\mathbf{h}^{-1}\mathbf{k_{a_1}}^{-1}$. But provided $(x_{a_2}, \mathbf{k_{a_2}})$ produces $\mathbf{P_{a_1}}$ we must find y_{a_2} such that $\mathbf{P_{a_1}} = \mathbf{k_{a_2}}\mathbf{g}\mathbf{d_u}^{x_{a_2}}\mathbf{g}^{-1}\mathbf{k_{a_2}}^{-1}$ and $\mathbf{Q_{a_1}} = \mathbf{k_{a_2}}\mathbf{h}\mathbf{d_w}^{y_{a_2}}\mathbf{h}^{-1}\mathbf{k_{a_2}}^{-1}$. Our strategy to derive $\mathbf{Q_{a_1}}$ consists in constructing $\mathbf{d_w}$ such that each exponentiation produces a different matrix. Therefore, we will eventually arrive to the exponent that produces $\mathbf{Q_{a_1}}$. Therefore, we require that $\mathbf{h}\mathbf{d_w}^{i}\mathbf{h}^{-1} \neq \mathbf{h}\mathbf{d_w}^{j}\mathbf{h}^{-1} \dots$ for $i \neq j$. Removing $\mathbf{h}$ both sides we have $\mathbf{d_w}^{i} \neq \mathbf{d_w}^{j} \dots$ which implies that each diagonal element satisfies the condition ${d_{w_i}}^{i} \neq {d_{w_i}}^{j} \bmod p \dots$ for $i = 1 \dots n$ where n is the matrix dimension. To surpass such requirement each diagonal element will be computed as 2^{μ_i} so that $2^{\mu_i} < p$ for $i = 1 \dots m$. Thus, each diagonal element of $\mathbf{d_w}$ is a power of the primitive root inside $\mathbb{Z}_p$.

In this enhanced scenario, the size of the public key yields $\mid (\mathbf{P_i}, \mathbf{Q_i}) \mid = 512$, the private key $\mid (x_i, y_i, \mathbf{k_i}) \mid = 512$ 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 $\mathbf{k_s} = \mathbf{u}^{x_a x_b}\mathbf{w}^{y_a y_b}$ defined in $\mathbb{Z}_p^{n\times n}$.

User	$\mathbf{P_i}$	$\mathbf{Q_i}$
Alice	$\mathbf{k_a}\mathbf{u}^{x_a}\mathbf{k_a}^{-1}$	$\mathbf{k_a}\mathbf{w}^{y_a}\mathbf{k_a}^{-1}$
Bob	$\mathbf{k_b}\mathbf{u}^{x_b}\mathbf{k_b}^{-1}$	$\mathbf{k_b}\mathbf{w}^{y_b}\mathbf{k_b}^{-1}$

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 certified keys from the web service. Then, they perform the usual exponentiation $(\mathbf{u}^{x_i x_{ca}})^{x_j}$. The secret shared key is derived as $\mathbf{u}^{x_i x_{ca} x_j}$.

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

User	Public key	Certified key
CA	$\mathbf{k_{ca}}\mathbf{u}^{x_{ca}}\mathbf{k_{ca}}^{-1}$	-
Alice	$\mathbf{k_a}\mathbf{u}^{x_a}\mathbf{k_a}^{-1}$	$\mathbf{k_a}\mathbf{u}^{x_a x_{ca}}\mathbf{k_a}^{-1}$
Bob	$\mathbf{k_b}\mathbf{u}^{x_b}\mathbf{k_b}^{-1}$	$\mathbf{k_b}\mathbf{u}^{x_b x_{ca}}\mathbf{k_b}^{-1}$

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 $\mathbf{k_i}\mathbf{u}^{x_i x_{ca_1} x_{ca_2}}\mathbf{k_i}^{-1}$ and $\mathbf{k_j}\mathbf{u}^{x_j x_{ca_1} x_{ca_2}}\mathbf{k_j}^{-1}$ of user i and j, respectively. The shared secret key between users will be $\mathbf{u}^{x_i x_{ca_1} x_{ca_2} x_j}$ 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 $\mathbf{k_t}$ based on the they already have $\mathbf{k_s}$. However, if for some reason $\mathbf{k_t}$ 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 $\mathbf{k_a}$ and $\mathbf{k_b}$ remain secret, the eavesdropper could capture $\mathbf{k_t}$ but she does not know $t_a t_b$ thus she cannot derive $\mathbf{k_s}$.

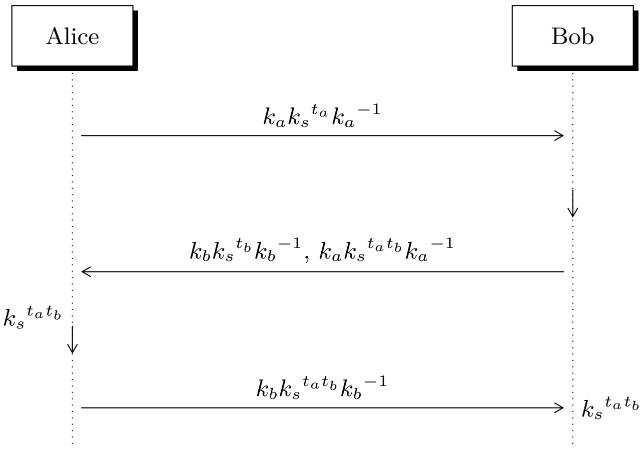

Fig. 4: Perfect Forward Secrecy (PFS) in Lizama's non-commutative Key Exchange Protocol (nc-KEP). The new shared secret key between users is $\mathbf{k_t} = \mathbf{k_s}^{t_a t_b}$ defined in $\mathbb{Z}_p^{n \times n}$. Symbols in the figure denote matrices.

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 2×2. A user's public key $\mathbf{P_i}$ is represented as $\mathbf{P_i} = \mathbf{k_i}\mathbf{u}^{x_i}\mathbf{k_i}^{-1}$ where $\mathbf{u}$ is a diagonalizable matrix that can be written as $\mathbf{g}\mathbf{d_u}\mathbf{g}^{-1}$, therefore $\mathbf{u}^{x_i} = \mathbf{g}\mathbf{d_u}^{x_i}\mathbf{g}^{-1}$, where $\mathbf{d_u}$ is the diagonal matrix. Then we have $\mathbf{P_i} = \mathbf{k_i}\mathbf{g}\mathbf{d_u}^{x_i}\mathbf{g}^{-1}\mathbf{k_i}^{-1}$ in $\mathbb{Z}_p^{n\times n}$. Now we proceed according to Equation 5.

$$\begin{aligned}
\mathbf{P_i} &= \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix} \begin{bmatrix} d_{11}{}^{x_i} & \\ & d_{22}{}^{x_i} \end{bmatrix} \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix}^{-1} \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix}^{-1} \bmod p \\
\mathbf{P_i} &= \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix} \begin{bmatrix} d_{11}{}^{x_i} & \\ & d_{22}{}^{x_i} \end{bmatrix} \begin{bmatrix} s_{11} & s_{12} \\ s_{21} & s_{22} \end{bmatrix} \begin{bmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{bmatrix} \bmod p
\end{aligned} \tag{5}$$

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

$$\begin{aligned}
P_{i_{11}} &\equiv (s_{11}t_{11}+s_{12}t_{21})(k_{11}g_{11}+k_{12}g_{21})d_{11}{}^{x_i} \bmod p + (s_{21}t_{11}+s_{22}t_{21})(k_{11}g_{12}+k_{12}g_{22})d_{22}{}^{x_i} \bmod p \\
P_{i_{12}} &\equiv (s_{11}t_{12}+s_{12}t_{22})(k_{11}g_{11}+k_{12}g_{21})d_{11}{}^{x_i} \bmod p + (s_{21}t_{12}+s_{22}t_{22})(k_{11}g_{12}+k_{12}g_{22})d_{22}{}^{x_i} \bmod p \\
P_{i_{21}} &\equiv (s_{11}t_{11}+s_{12}t_{21})(k_{21}g_{11}+k_{22}g_{21})d_{11}{}^{x_i} \bmod p + (s_{21}t_{11}+s_{22}t_{21})(k_{21}g_{12}+k_{22}g_{22})d_{22}{}^{x_i} \bmod p \\
P_{i_{22}} &\equiv (s_{11}t_{12}+s_{12}t_{22})(k_{21}g_{11}+k_{22}g_{21})d_{11}{}^{x_i} \bmod p + (s_{21}t_{12}+s_{22}t_{22})(k_{21}g_{12}+k_{22}g_{22})d_{22}{}^{x_i} \bmod p
\end{aligned}$$

As can be seen there, each term of the public key depends on a number of secret integers due to matrix multiplication. For example, $P_{i_{11}}$ depends on k_{11}, k_{12}, t_{11} and t_{21} which are kept secret. Provided the elements of $\mathbf{k_i}$ remain private, so the elements of $\mathbf{k_i}^{-1}$.

We do not devise a method to derive the private key $(x_i, \mathbf{k_i})$ from $\mathbf{P_i}$ 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 $\mathbf{k}$ and x, thus infeasible for practical purposes. As long as the total size of $\mathbf{k}$ is at least 256 bits, the method is assumed to be resistant to the Grover's quantum search [26].

References

1. M. A. Barreno, "The future of cryptography under quantum computers," *Dartmouth College Computer Science Technical Reports*, 2002.
2. P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th annual symposium on foundations of computer science*, pp. 124–134, Ieee, 1994.
3. H. Bennett Ch and G. Brassard, "Quantum cryptography: public key distribution and coin tossing int," in *Conf. on Computers, Systems and Signal Processing (Bangalore, India, Dec. 1984)*, pp. 175–9, 1984.
4. L. A. Lizama-Pérez, J. M. López, and E. D. C. López, "Quantum flows for secret key distribution," *Advanced Technologies of Quantum Key Distribution*, p. 37, 2018.
5. L. A. Lizama-Perez and J. M. López, "Quantum key distillation using binary frames," *Symmetry*, vol. 12, p. 1053, Jun 2020.
6. L. A. Lizama-Pérez and J. M. López R., "Beyond the limits of shannon's information in quantum key distribution," *Entropy*, vol. 23, no. 229, 2021.
7. C. S. R. CENTER, "Post-Quantum Cryptography Standardization Conference," 2021. [Online; accessed May 10, 2021].
8. L. A. Lizama-Perez, "Digital signatures over hash-entangled chains," *SN Applied Sciences*, vol. 1, no. 12, p. 1568, 2019.
9. L. A. Lizama-Pérez, L. J. Montiel-Arrieta, F. S. Hernández-Mendoza, L. A. Lizama-Servín, and S.-A. Eric, "Public hash signature for mobile network devices," *Ingeniería, Investigación y Tecnología*, vol. XX, no. 2, pp. 1–10, 2019.
10. D. J. Bernstein, "Introduction to post-quantum cryptography," in *Post-quantum cryptography*, pp. 1–14, Springer, 2009.

 doi:10.20944/preprints202105.0174.v1

11. I. T. Laboratory, “PQC Standardization Process: Third Round Candidate Announcement.” `https://csrc.nist.gov/news/2020/pqc-third-round-candidate-announcement`, 2020. [Online; accessed May 10, 2021].
12. L. Chen, L. Chen, S. Jordan, Y.-K. Liu, D. Moody, R. Peralta, R. Perlner, and D. Smith-Tone, *Report on post-quantum cryptography*, vol. 12. US Department of Commerce, National Institute of Standards and Technology, 2016.
13. E. Persichetti, “NIST Round 3 finalists.” `https://pqc-wiki.fau.edu/w/Special:DatabaseHome`, 2020. [Online; accessed May 10, 2021].
14. L. A. Lizama-Pérez and J. M. López R., “Non-invertible public key certificates,” *Entropy*, vol. 23, no. 2, 2021.
15. W. Diffie and M. Hellman, “New directions in cryptography,” *IEEE transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, 1976.
16. D. Kahrobaei, C. Koupparis, and V. Shpilrain, “Public key exchange using matrices over group rings,” *arXiv preprint arXiv:1302.1625*, 2013.
17. E. Stickel, “A new method for exchanging secret keys,” in *Third International Conference on Information Technology and Applications (ICITA’05)*, vol. 2, pp. 426–430, IEEE, 2005.
18. A. Myasnikov, V. Shpilrain, and A. Ushakov, *Group-based cryptography*. Springer Science & Business Media, 2008.
19. V. Shpilrain, “Cryptanalysis of stickel’s key exchange scheme,” in *International Computer Science Symposium in Russia*, pp. 283–288, Springer, 2008.
20. D. Grigoriev and V. Shpilrain, “Tropical cryptography,” *Communications in Algebra*, vol. 42, no. 6, pp. 2624–2632, 2014.
21. J. Ding, X. Xie, and X. Lin, “A simple provably secure key exchange scheme based on the learning with errors problem.,” *IACR Cryptol. ePrint Arch.*, vol. 2012, p. 688, 2012.
22. L. A. Lizama-Perez and J. M. López, “Quantum key distillation using binary frames,” *Symmetry*, vol. 12, no. 6, p. 1053, 2020.
23. L. A. Lizama-Perez, “Non-invertible key exchange protocol,” *SN Applied Sciences*, vol. 2, p. 1083, 2020.
24. J. H. Cheon and B. Jun, “A polynomial time algorithm for the braid diffie-hellman conjugacy problem,” in *Annual International Cryptology Conference*, pp. 212–225, Springer, 2003.
25. C. Zalka, “Grover’s quantum searching algorithm is optimal,” *Physical Review A*, vol. 60, no. 4, p. 2746, 1999.
26. L. K. Grover, “A fast quantum mechanical algorithm for database search,” in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pp. 212–219, ACM, 1996.