entropy

MDPI

Article

Non-Invertible Public Key Certificates

Luis Adrián Lizama-Perez [1,*] and J. Mauricio López R. [2]

[1] Dirección de Investigación, Innovación y Posgrado, Universidad Politécnica de Pachuca, Ex-Hacienda de Santa Bárbara, Zempoala, Hidalgo 43380, Mexico
[2] Cinvestav Querétaro, Libramiento Norponiente 2000, Real de Juriquilla, Santiago de Querétaro, Querétaro 76230, Mexico; jm.lopez@cinvestav.mx
* Correspondence: luislizama@upp.edu.mx

Abstract: Post-quantum public cryptosystems introduced so far do not define a scalable public key infrastructure for the quantum era. We demonstrate here a public certification system based on Lizama's non-invertible key exchange protocol which can be used to implement a secure, scalable, interoperable and efficient public key infrastructure (PKI). We show functionality of certificates across different certification domains. Finally, we discuss a method that enables non-invertible certificates to exhibit perfect forward secrecy (PFS).

Keywords: non-invertible; cryptography; certificate; PKI

Citation: Lizama-Perez, L.A.; López R., J.M. Non-Invertible Public Key Certificates. *Entropy* **2021**, *23*, 226. http://doi.org/10.3390/e23020226

Academic Editor: Cosmo Lupo, Rupesh Kumar and Alessandro Zavatta
Received: 18 December 2020
Accepted: 10 February 2021
Published: 12 February 2021

1. Introduction

Since its origin in the late seventies, public key cryptography (PKC) has been exploited to support user authentication and digital signatures over the internet. In PKC, each user has two keys, the public P_u and the private key P_r, which are mutually inverse in some mathematical sense. Not taking into account formal details we would write that $P_r = {P_u}^{-1}$ thus, to achieve confidentiality, a message m is encrypted using Bob's public key; symbolically we write $[m]_{P_u}$, then it is decrypted with the private key so $m = [m]_{P_u {P_u}^{-1}}$. In contrast, to guarantee message authentication, m is encrypted with Alice's private key and decrypted with her public key. Symbolically we can write it as $m = [m]_{P_r {P_r}^{-1}}$.

Unfortunately, Shor's algorithm [1] solves over a hypothetical quantum computer, the mathematical problems on which PKC is supported: integer factorization and discrete logarithm. In fact, most of the public key cryptosystems used today will become obsolete in the foreseeable future because they would be broken by quantum computers [2]. For this reason, the National Institute of Standards and Technology (NIST) initiated in 2015 a process to evaluate cryptographic algorithms to choose the appropriate methods for the quantum era. To this date, the selection process is in the third evaluation round [3,4].

The present work enhances a newly claimed post-quantum method called non-invertible key exchange method (ni-KEP) which was conceived to establish a secret key between two remote parties. Lizama's ni-KEP is mathematically supported by Euler's theorem as RSA, it uses exponentiation to exchange a secret key as Diffie–Hellman and it encrypts/decrypts through invertible multiplication as ElGamal cipher. Lizama's non-invertible key exchange protocol was introduced in [5]. Initially, the protocol was conceived to transfer a secret value from Alice to Bob. We describe briefly the three development stages of the algorithm:

1. *Multiplication-based protocol.* In a ring with unity over $\mathbb{Z}_n$ where $n = p \cdot q$ and p, q are prime numbers. An integer may or may not have a multiplicative inverse. Multiplication between invertible and a non-invertible integer yields a non-invertible integer according to the basic properties of modular arithmetic. Alice multiplies a random non-invertible v_a by a random invertible k_a, then she sends the result to Bob who multiplies it by his random invertible k_b returning the resulting integer

to Alice who removes k_a multiplying by k_a^{-1} and sending the result to Bob. Finally, Bob removes his invertible integer applying k_b^{-1}. At this point Bob has obtained v_a. Although a non-invertible integer does not have a multiplicative inverse, hence factorization of the public integers are prohibited, a division attack is discussed in [5].

2. *Exponent-based protocol.* The integer that results after exponentiation say p^{x_a} gives a non-invertible integer. Using this math property, the protocol defines that Alice sends $p^{x_a}k_a \bmod n$ to Bob who returns $p^{x_a}k_ak_b \bmod n$ to her. Then she multiplies it by k_a^{-1} and sends back $p^{x_a}k_b \bmod n$. Bob applies k_b^{-1} thus obtaining the shared secret $p^{x_a} \bmod n$. Unfortunately, this version of the protocol is also vulnerable to a division attack [5].
3. *Non-invertible KEP.* This protocol defines a public key exchange algorithm. To surpass the division attack, ni-KEP introduces Euler's identity to derive the keys which are defined according to the relations $\{p^{x_i}k_i \bmod n,\ q^{y_i}k_i \bmod n\}$, $i = a, b$ for Alice's and Bob's public keys respectively and n is obtained as $n = p \cdot q \cdot r$ where p and q are small prime public numbers and r is a big prime public integer. On the other hand, $\{k_i, x_i\}$ constitute the private key, while the number y_i is derived from the equation $\phi(n) = x_i + y_i + 1$ where $\phi(n)$ is the Euler's totient equation. A detailed discussion of this protocol will be presented in a later section.

The public keys of the ni-KEP (and also the cipher texts) exhibit perfect indistinguishability [5]. It means, in the first case, that every k_i in the ring satisfies the public key (exists a corresponding x_i). In the second case, it implies that each ciphertext c_i can be derived by any k_i in the ring (exists a corresponding m_i). In view of the above, we claim that the unique opportunity for the eavesdropper, in order to get the private key (or the plaintext), is implementing an exhaustive search among the elements in the ring which is equivalent to searching an unsorted database problem.

Consider symmetric cryptography, which is assumed to be post-quantum because a quantum computer running Grover's algorithm requires computational cost proportional to the square root of the key size which takes $O(\sqrt{N})$ time. Despite this, an adjustment in the key size prevents the crypto system of being vulnerable to Grover's algorithm which is the fastest possible quantum algorithm for searching an unsorted database. By contrast, a classical computer requires a linear search, which is $O(N)$ in time to find the same entry [6].

For this reason, we claim that our algorithm is post quantum. On the other hand, we do not devise how Shor's algorithm would be used to break this protocol. As a consequence, Lizama's key sizes must be carefully chosen to resist a hypothetical quantum computer running Grover's algorithm.

Our contribution. In this work, we enhance Lizama's non-invertible key exchange method [5] in order to support Certification Authorities (CA) to allow users to exchange digital certificates which are bounded to their public keys. We claim that our cryptosystem exhibits competitive key size and is able to handle certificated keys, interdomain certification and perfect forward secrecy.

Organization of the paper. First, in Section 2, we discuss the main quantum cryptographic approaches: quantum and post-quantum. In Section 3, we summarize principles of public key cryptography considering digital certificates and the Certification Authority role. Then we describe in Section 4 Lizama's non-invertible protocol to put forward and in Section 5 how Lizama's KEP can be used to support CAs in single and multiple certification domains. Finally, Section 6 explains a method to derive a new session key from a past session key, thus achieving Perfect Forward Secrecy (PFS). Appendix A contains a brief description about RSA and DH cryptosystems along two possible attacks: prefix and multiplication-based attacks.

2. Cryptography in the Quantum Era

Cryptography in the quantum era can be classified into two main approaches: quantum and post-quantum cryptography. A formal discussion of such approaches is beyond the scope of the present article. Let us simply mention that quantum cryptography relies

on quantum physics principles that allow to establish a secret key between two authenticated remote parties [7]. The eavesdropper cannot control quantum communication because it produces a detectable noise. Works have been done recently to resist quantum attacks [8–10].

On the other side, post-quantum cryptography encompasses cryptographic mathematical methods conceived to resist computational capacity of quantum computers [4,11]. Several methods have been formulated based on computational problems whose complexity surpass the theoretical capacities of quantum computers. Not wishing to fully cover all cases, most promissory techniques include lattices, supersingular isogeny, multivariate equations, code and hash-based cryptography.

Lattice-based methods have demonstrated good performance, by generating short ciphertext, short keys and short signatures [12,13]. Similar to Diffie–Hellman key exchange is the Supersingular Isogeny Diffie–Hellman (SIDH) method which is a quantum resistant key exchange algorithm [14,15]. Supersingular Elliptic Curve Isogeny Cryptography (SIDH) produces very small key sizes but it shows slower performance. The representative algorithm is the Supersingular Isogeny Key Encapsulation (SIKE). The basic objects of multivariate cryptography are systems of nonlinear (usually quadratic) polynomial equations in several variables over a finite field. When performing a digital signature, the set of equations constitute the public key. The receiver computes the hash to verify that the output of the equations corresponds to the hash of the message that is signed [16]. A code-based cryptosystem is essentially a form of error correction code. The private key is a code C, which allows to correct t errors. The sender will encode the message with the public key and include t errors during encoding, then the ciphertext is obtained by adding an error vector to each codeword. With code C, the receiver will be able to accurately correct the errors when decoding the message. Hash-based cryptography was introduced by Lamport, later it was enhanced using Merkle trees [17] and Lizama's hash-based methods [18,19].

3. Public Key Cryptography

3.1. Digital Certificates

A cryptographic certificate is basically, a verified public key signed by a third trusted party called Certification Authority (CA). By using this method, each user can verify the origin of a request before accepting it. The importance of a certified key can be illustrated showing a man in the middle (MITM) attack over the Diffie–Hellman (DH) protocol, the first public key exchange algorithm [20]. In Figure 1, we represent the steps required for this key exchange algorithm where the integer prime p and g are publicly known. A description in depth can be found in Appendix A.

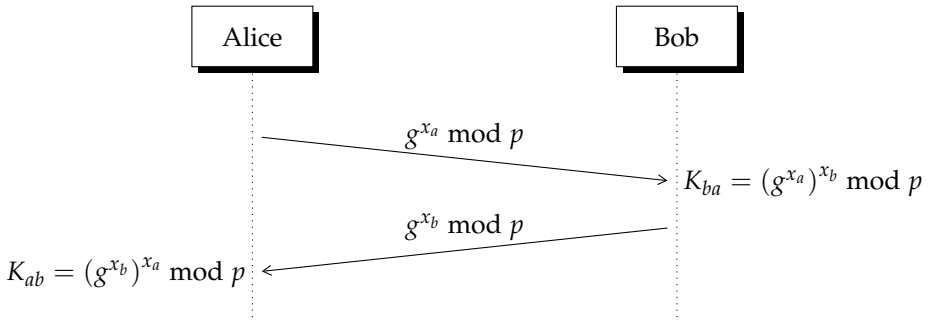

Figure 1. Basic Diffie–Hellman protocol. All operations are performed module p.

Since there is no method to verify the origin of the integer numbers exchanged across the public channel, an eavesdropper can implement a man in the middle (MITM) attack over the Diffie–Hellman method as it is observed in Figure 2.

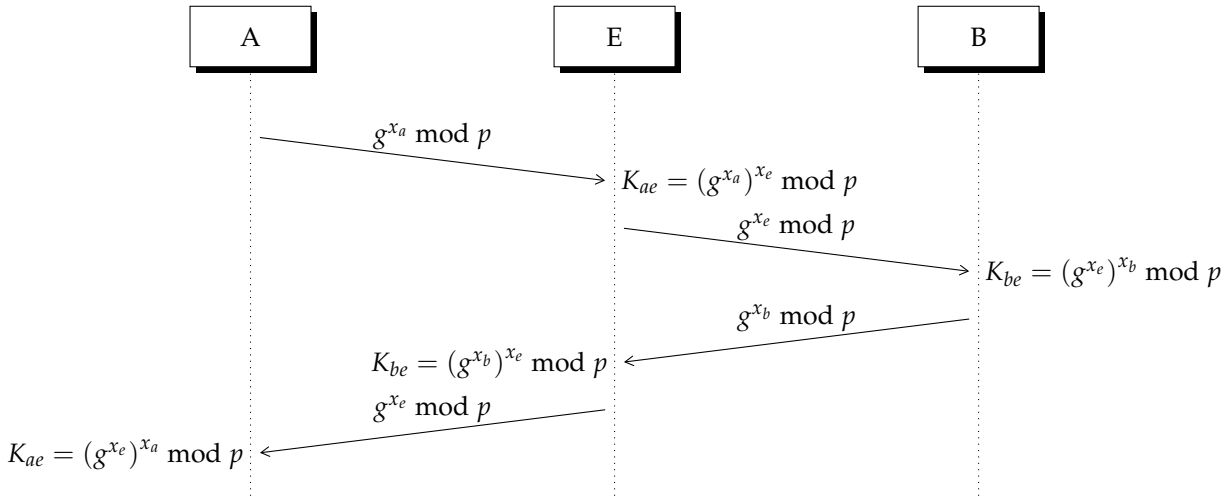

Figure 2. A man in the middle (MITM) attack over Diffie–Hellman (DH) protocol. The eavesdropper obtains a key with Alice K_{ae} and other with Bob K_{be}. Legitimate users cannot verify the origin of exchanged numbers.

To avoid an MITM attack over DH protocol, the RSA algorithm can be added to the exchange protocol. RSA is described in Appendix A. Another common method to protect DH key exchange algorithm is elliptic-curve cryptography [21,22], however Lizama's protocol is closely related to RSA, thus we describe here RSA and DH.

Figure 3 shows that Alice encrypts the DF constructor $g^{x_a} \bmod p$ with Bob's public key written as (e_b, n_b), so that only Bob can decrypt it using his private key d_b. Alice verifies the received message because it is attached a hash of the secret key computed by Bob as represented in Figure 3.

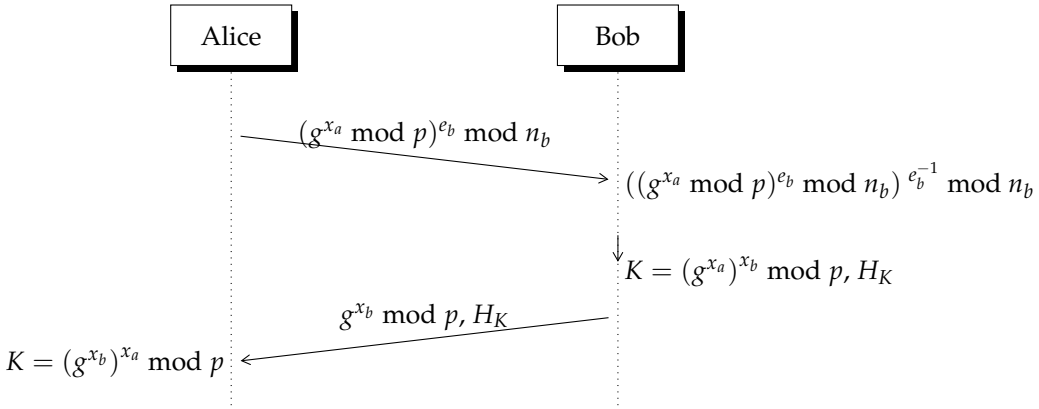

Figure 3. Diffie–Hellman algorithm with RSA. Bob's public key is written as $P_{U_B} = (e_b, n_b)$, Bob's private key is ${e_b}^{-1}$ that indicates the inverse of e_b in $\mathbb{Z}_{\phi(n)}$. H_K represents the hash value of K which is used by Alice to verify the origin of the received number.

In order for Alice to verify Bob's public key, provided it does not come from an illegitimate user, Bob must register first his public key with the Certification Authority abbreviated as CA (a third trusted party). Generally speaking, Bob obtains a certificate of his public key C_B after CA encrypts (signing) Bob's public key with CA's private key $P_{R_{CA}}$. In the next relations, encryption (or decryption) process is denoted as square brackets while the encryption (or decryption) key is outside the brackets:

$$C_B = [P_{U_B}]_{P_{R_{CA}}}$$

Every user can obtain and verify Bob's public key decrypting C_B with CA's public key $P_{R_{CA}}$:

$$P_{U_B} = [C_B]_{P_{U_{CA}}}$$

3.2. Certification Authority (CA)

As mentioned earlier, a Certification Authority (CA) is a trusted third party that signs a user public key using CA's private key therefore binding the subject's identity (and associated information including the name of the owner) to the user's public key inside a cryptographic certificate. Cryptographic certificates can be exploited to achieve digital signatures in a wide broad of internet transactions and PKI: certificates (X.509), secure channels (TLS) and email (S/MIME).

In view of the imminent arrival of quantum computers, it is unpostponable to develop strategies to adapt the public key infrastructure (PKI) for transition to the quantum era [3,4]. Up to now, few works have been published that adapt existing certificates to quantum certificates or hybrid certificates, which include two public keys for the subject, one classical and one post-quantum and two CA signatures [23,24]. Other works have evaluated existing mechanisms to deal with large records like record fragmentation, segmentation, caching and compression [25]. One the main challenges reported is the difficulty to manage larger certificates by some cryptographic software libraries.

ITU-T Recommendation X.509 defines the format of public key certificates as well as the provision of authentication services under a centralized control scheme that is represented by a directory [26,27]. X.509 assumes a hierarchical system of Certificate Authorities (CAs) for issuing certificates. This contrasts with web of trust models, like PGP, where users sign others' key certificates to establish the authenticity of the binding between a public key and its owner [28].

A PKI is arranged hierarchically, so that there is always a direct path (a certificate chain) from the root CA to every end-entity. Therefore, with many users, it may be more practical to have a series of CAs, each of which securely provides its public key to a fraction of the users.

If Alice has a certificate from CA_1 and Bob owns a certificate from CA_2 but Alice does not securely know the public key of CA_2, then Bob's public certificate emitted by CA_2, cannot be used by Alice. However, if the two CAs have securely exchanged their own public keys, the following procedure will enable Alice to obtain Bob's public key:

1. Alice obtains the certificate of CA_2 signed by CA_1. Since Alice has the public key of CA_1, she can get the public key of CA_2 from its certificate and verify it using the signature of CA_1 on the certificate.
2. From the directory, Alice obtains the certificate of Bob signed by CA_2. Since Alice now has the public key of CA_2, she can verify the signature, therefore getting Bob's public key.

4. Lizama's Key Exchange Protocol

Lizama's key exchange protocol was introduced in [5], there it can be found all details about the method and its security. The protocol is illustrated in Figure 4. The public key of user i (a for Alice, b for Bob) 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 $y_i = \phi(n) - x_i + 1$. The module n is the product of tree public integer primes, so that $n = p \cdot q \cdot r$ where p and q are small integer primes and r is a big integer prime. To achieve indistinguishability p and q are suggested to be 2, since 2 is a primitive root module r (see [5]). The exponent is chosen to be $2x_i$ instead of x_i to avoid a multiplication attack (see Appendix A). The x_i value constitutes along k_i the private key of user i where k_i is an invertible integer in the ring. Users share their public keys (P_a, Q_a) and (P_b, Q_b) as well as the integer module n. The steps of the protocols are summarized as follows:

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

Table 1. These operations (exponentiation and multiplication) are performed at each side after public keys of users are exchanged.

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

2. To derive the results in the right column of Table 1, Euler's theorem is applied in $\mathbb{Z}_n$. The theorem is written in Equation (1) where r is an integer safe prime. As a result that $n = pqr$, we have that $\phi(n) = (p-1)(q-1)(r-1)$. Here, k and n are relative prime to each other, so k is an invertible integer in $\mathbb{Z}_n$. The exponent x_i constitutes the private key, is chosen randomly, but x_i and y_i sum up $\phi(n)+1$, thus according to Equation (1) we have $k^{\phi(n)+1} = k^{\phi(n)} \cdot k^1 = k$ because k is an invertible integer in $\mathbb{Z}_n$.

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

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 4.

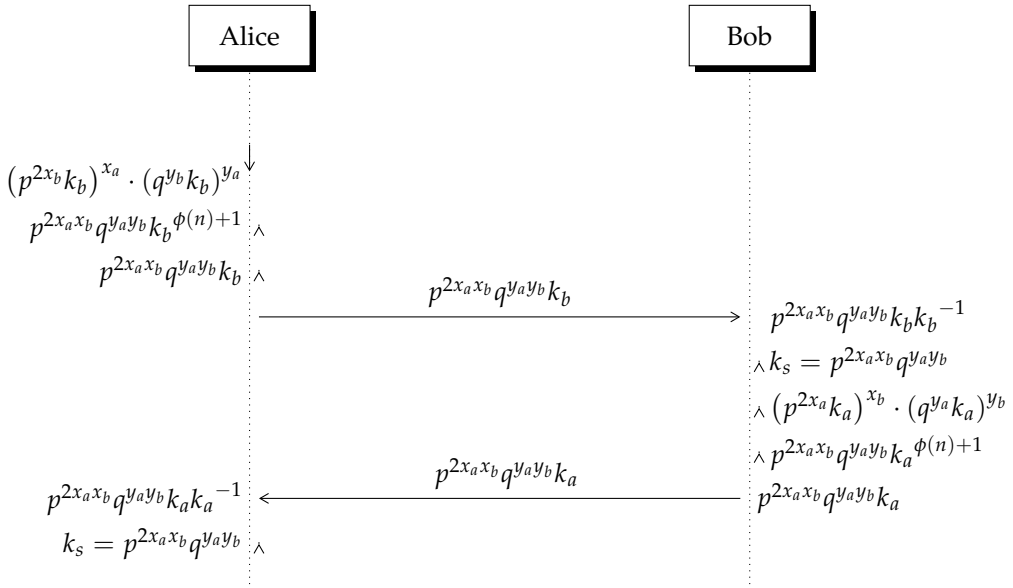

Figure 4. Lizama's non-invertible key exchange method (KEP) [5]. All operations are modulo n where $n = pqr$. According to Euler's theorem $k^{\phi(n)+1} \bmod n = k$ because k is an invertible integer in $\mathbb{Z}_n$.

As an example of the required bits for the keys, consider that case where $p = q = 2$ and $|r| = 1024$ (the symbol $|\ |$ denotes the number of bits) the length of the private key yields 1536 bits ($|x| = 512$ and $|k| = 1024$) while the public key (P, Q) contains 2056 bits [5]. In this case, the security level of the secret key is 1024. The process to determine the size of the key is the following: $P = p^{2x} \cdot k \bmod n$ thus $P = p^{2x} \bmod n \cdot k \bmod n$, which in turn implies that:

— $|k| = |n|$
— if $p = 2$ and $n = 4r$, we have $2^{2x} \bmod 4r$, then $4^x \bmod 4r$ yields $|4| \cdot |x| = |4| + |r|$ and $|x| \sim \frac{|r|}{2}$.
— since the private key is conformed by x and k, its size is computed as $|n| + |x| \sim |r| + |x|$ which gives 1536.

4.1. Cipher-System

In Figure 4, the secret shared key k_s is a non-invertible number in $\mathbb{Z}_n$, thus a convenient method to achieve a cipher-system and secret communication is to divide $k_s = p^{2x_a x_b} q^{y_a y_b} \mod n$ by pq, so if we choose $p = q = 2$, then $k_r = p^{2x_a x_b - 2} 2^{(2r-1-x_a)(2r-1-x_b)} \mod r$. Now, Alice and Bob can compute its multiplicative inverse ${k_r}^{-1}$. Table 2 shows that the enciphered message is obtained as $c = m \cdot k_r \mod r$ and the original plaintext is recovered through the relation $m = c \cdot {k_r}^{-1} \mod r$ because $m = m \cdot k_r {k_r}^{-1} \mod r$. To send a message encoded as an integer in $\mathbb{Z}_r$, the number m must be less than r.

Table 2. Lizama's key exchange algorithm can be used to encrypt/decrypt messages provided k_s is divided by pq.

Mode	Mathematical Relation
Encryption	$c = m \cdot k_r \mod r$
Decryption	$m = c \cdot {k_r}^{-1} \mod r$

4.2. Mathematical Representation

In the rest of the paper we will use the following mathematical notation: (P_i, Q_i) which constitutes the public key of user i. As stated before, $P_i = p^{2x_i} k_i$ and $Q_i = q^{y_i} k_i$ where (x_i, k_i) constitutes the private key of user i and $x_i + y_i = \phi(n) + 1$. As stated before, user j raises the public key of i to its private key. Then, j returns to i the integer number $[k_{i,j}]\ k_i$ where $[k_{i,j}] = p^{2x_i x_j} q^{y_i y_j}$ and k_i is a component of the private key of user i, then they apply the inverse of k_i in order to derive the shared secret key $k_{i,j}$. The same procedure is applied in the opposite direction so user i sends to j the integer $[k_{i,j}]\ k_j$ to get the secret number $k_{i,j}$ (see Table 3).

Table 3. Mathematical representation. All operations are performed module n.

Short Notation	Mathematical Operation
(P_i, Q_i)	$P_i = p^{2x_i} k_i,\ Q_i = q^{y_i} k_i$
${P_i}^{x_j} \cdot {Q_i}^{y_j}$	$\left(p^{2x_i} k_i\right)^{x_j} \cdot \left(q^{y_i} k_i\right)^{y_j}$
$[k_{i,j}]\ k_i$	$p^{2x_i x_j} q^{y_i y_j} k_i$

5. Key Certification with Lizama's ni-KEP

In this section, we explain the public key certification method so that a Certification Authority (CA) can certify the user's public keys using Lizama's ni-KEP. The protocol steps are as follows:

1. To certify their public key with the Certification Authority CA, user i sends to CA their public key (P_i, Q_i).
2. If CA approves the request of i, they generate and publish the certified key $[k_{i,ac}]\ k_i$ which has been derived according to Table 3.
3. The CA's public database of certified keys can be seen in Table 4 which contains the certified keys of Alice and Bob.

Table 4. CA's public database. The Certification Authority CA publishes their public key (P_{ca}, Q_{ca}).

User	Public Key	Certified Key
CA	(P_{ca}, Q_{ca})	-
Alice	(P_a, Q_a)	$[k_{a,ca}]\ k_a$
Bob	(P_b, Q_b)	$[k_{b,ca}]\ k_b$

Now, Alice and Bob can establish a secret key with certified keys, but first Alice must download Bob's certified key from CA's database and vice versa. The steps to derive the key are depicted in Figure 5 and described as follows:

1. Using CA's public key (P_{ca}, Q_{ca}), Alice computes $[k_{a,ca}]\ k_{ca}$. In addition, she computes $[k_{a,b}]\ k_b$ using Bob's public key (P_b, Q_b).
3. Alice multiplies them by Bob's certified key $[k_{b,ca}]k_b$ and sends the resulting integer number to Bob. The same procedure is applied by him.
4. Bob multiplies the received integer by ${k_b}^{-1}$ twice, thus he obtains the secret shared key $K_{ab} = [k_{a,b}][k_{b,ca}][k_{a,ca}]k_{ca}$ (see Figure 5).
5. Applying this procedure, Bob derives the same secret number K_{ab}.

It must be highlighted that in order to establish the secret key the certified key of the intended user must be applied but also the public key of the Certification Authority CA. Moreover, each user must apply (twice) their private key to get the shared secret key. In addition, to avoid a prefix attack the relation $K_{ab} > r$ must be satisfied (see Appendix A).

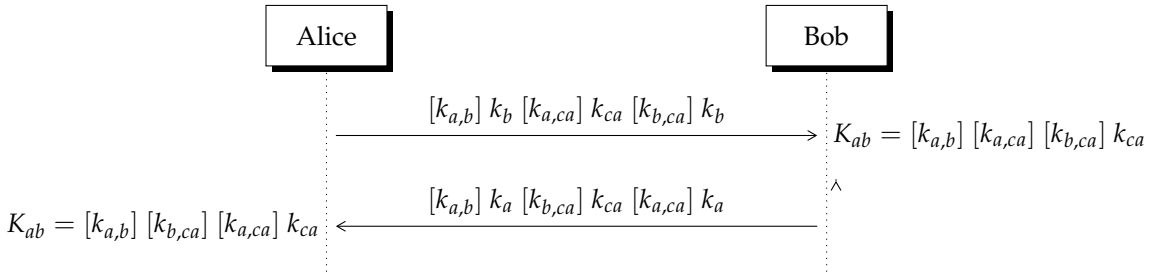

Figure 5. Non-invertible KEP with Certification Authority (CA). All operations are performed module n.

5.1. Indistinguishability

An important security property of the ni-KEP is the indistinguishability of k_i in the public key integers. It implies that each invertible k_i in $\mathbb{Z}_n$ satisfies the public key along the appropriate x_i value. The same property can be deduced for the cipher text, thus every k_i in the ring can be used to produce a given cipher text with a specific m_i.

Indistinguishability can be extended to the certified key exchange method. Let us rewrite the exchanged messages depicted in Figure 5 as $M_b \cdot k_b \bmod m$ from Alice to Bob so that $M_b = [k_{a,b}]k_b[k_{a,ca}]k_{ca}[k_{b,ca}]$. Similarly, in the reverse direction we have $M_a \cdot k_a \bmod n$ which implies that $M_a = [k_{a,b}]k_a[k_{b,ca}]k_{ca}[k_{a,ca}]$. Applying division to M_a (or M_b) by pq we obtain:

$$(pq)^{-1} M_i \cdot k_i \bmod r$$

From here, we know that $M_i \bmod r$ and $k_i \bmod r$ are integers in $\mathbb{Z}_r$. Moreover, the multiplication $M_i \cdot k_i \bmod r$ produces a permutation of the integers in $\mathbb{Z}_r$ because r is an integer prime, thus the resulting integer is in $\mathbb{Z}_r$. As it was shown in [5], k_i remains indistinguishable inside encrypted messages; therefore, the unique opportunity for the eavesdropper is to find the secret key k_i by exhaustive search.

5.2. Multiple CAs

Suppose Alice has been registered with CA_1 while Bob has a certified key from CA_2. In addition, Alice receives from Bob its certified key and vice versa but Alice does not have access to CA_2's database neither Bob to CA_1's database. As indicated in Table 5, CA_1's database is accessible to Alice and CA_2's database is reachable by Bob. However, as can be seen there, CA_1's database contains the certified key of CA_2 and CA_2's database contains the certificate of CA_1. Then, they follow the steps depicted in Figure 6 and detailed below:

1. Using CA_1's public key (P_{ca_1}, Q_{ca_1}), Alice computes $[k_{a,ac_1}]\ k_{ac_1}$, she also computes $[k_{a,b}]\ k_b$ with Bob's public key (P_b, Q_b).
3. Alice multiplies them by Bob's certificate $[k_{b,ca_2}]k_b$ and CA_2's certificate $[k_{ca_1,ca_2}]\ k_{ca_2}$ and sends the resulting integer number to Bob. The same procedure is applied by Bob.
4. Alice multiplies the received integer by ${k_a}^{-1}$ twice, thus she obtains the secret shared key $K_{ab} = [k_{a,b}][k_{a,ca_1}]k_{ca_1}[k_{b,ca_2}]k_{ca_2}[k_{ca_1,ca_2}]$ (see Figure 6).

5. Applying the same procedure, Bob derives the secret shared number K_{ab}.

Table 5. Public databases of CA_1 and CA_2 which would be located distantly, so database of CA_1 is accessible to Alice and CA_2's database is close to Bob.

CA	User	Public Key	Certified Key
	CA_1	(P_{ca_1}, Q_{ca_1})	-
	CA_2	(P_{ca_2}, Q_{ca_2})	$[k_{ca_1,ca_2}]\ k_{ca_2}$
CA_1	Alice	(P_a, Q_a)	$[k_{a,ca_1}]\ k_a$
	CA_2	(P_{ca_2}, Q_{ca_2})	-
	CA_1	(P_{ca_1}, Q_{ca_1})	$[k_{ca_1,ca_2}]\ k_{ca_1}$
CA_2	Bob	(P_b, Q_b)	$[k_{b,ca_2}]\ k_b$

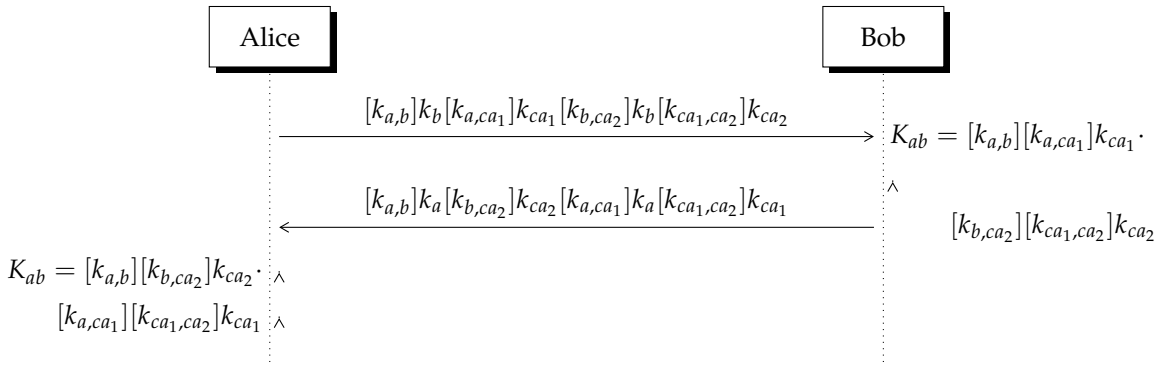

Figure 6. Non-invertible KEP with two CAs. Operations are performed module n.

6. Perfect Forward Secrecy (PFS)

Suppose Alice and Bob require to establish a new confidential communication. However, they do not want to use the same secret key of the last session. Perfect forward secrecy (PFS) is a feature of key agreement protocols that guarantee that, if the currently key was compromised, it does not compromise the security of previously used keys. Therefore, the security of encrypted messages using old keys persists. When a system has a perfect forward secret, the system is said to be forward secure.

In the next procedure, we demonstrate that Lizama's non-invertible KEP is enhanced to exhibit PFS (see Table 6 and Figure 7).

1. Alice and Bob share a certified key K_i from a previous exchange.
2. Using CA's public key (P_{ca}, Q_{ca}), Alice computes $[k_{a,ca}]\ k_{ca}$. In addition, according to Table 6, Alice computes $[k_{a,b}]^{K_i}\ {k_b}^{K_i}$ using Bob's public key (P_b, Q_b).
4. Alice multiplies them by Bob's certificate $[k_{b,ca}]\ k_b$ and sends the resulting number to Bob. The same procedure is applied by Bob.
5. Bob multiplies the received integer by ${k_b}^{-K_i-1}$, thus he obtains the secret shared key $K_{i+1} = [k_{a,b}]^{K_i}[k_{a,ca}][k_{b,ca}]k_{ca}$ (see Figure 7).
6. Conversely, Alice multiplies the received integer by ${k_a}^{-K_i-1}$, thus she gets the secret shared key $K_{i+1} = [k_{a,b}]^{K_i}[k_{b,ca}][k_{a,ca}]k_{ca}$.

Therefore, the eavesdropper cannot derive K_i from K_{i+1} and the procedure can be repeated as many times as required to derive K_{m+1} from K_m.

Table 6. Mathematical operations to achieve perfect secrecy (PFS).

Short Notation	Mathematical Operation
(P_i, Q_i)	$P_i = p^{2x_i}k_i$, $Q_i = q^{y_i}k_i$
${P_i}^{x_j} \cdot {Q_i}^{y_j}$	$(p^{2x_i}k_i)^{x_j} \cdot (q^{y_i}k_i)^{y_j}$
$[k_{i,j}]\ k_i$	$p^{2x_ix_j}q^{y_iy_j}k_i$
${P_i}^{k_sx_j} \cdot {Q_i}^{k_sy_j}$	$(p^{2x_i}k_i)^{k_sx_j} \cdot (q^{y_i}k_i)^{k_sy_j}$
$[k_{i,j}]^{k_s}\ {k_i}^{k_s}$	$p^{2k_sx_ix_j}q^{k_sy_iy_j}{k_i}^{k_s}$

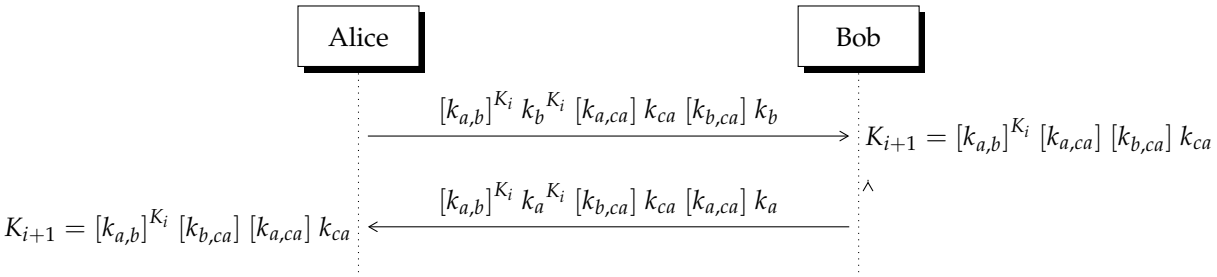

Figure 7. Alice and Bob require to establish a new secret key K_{i+1}. However, they do not want to use the last secret key K_i. This procedure is repeated to derive K_{i+2} from K_{i+1}.

7. Discussion

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. Since we are only concerned with the first category, we found that public keys size in Lizama's protocol has the smallest size: 0.256 kilobytes (for $|n| = 1024$) while the corresponding certified key size achieves 0.384 kilobytes (see Table 7). Furthermore, to reduce the required storage space a good strategy would be saving only one component of the public key (P_i, Q_i), e.g., P_i while the second one Q_i will be transferred directly from Alice to Bob. In that case the certified public key size decreases from 0.384 to 0.256 KB.

Table 7. A comparison of Lizama's protocol against National Institute of Standards and Technology (NIST) Round 3 finalists is shown in the categories of public key encryption and key-establishment algorithms [29].

Scheme	System	Public Key (KB)	Private Key (KB)	Signature (KB)
Public Key/ KEM	LIZAMA'S KEP	0.256–0.512	0.192–0.384	–
	Classic McEliece	261,120–1,357,824	6492–14,120	–
	CRYSTALS-KYBER	1.632–3.168	0.8–1.568	–
	NTRU-HPS	0.931–1.230	1.235–1.592	–
	SABER	0.672–1.312	1.568–3.040	–
Signature Algorithms	CRYSTALS-DILITHIUM	1.312–2.592	–	2.420–4.595
	FALCON	0.897–1.793	–	0.666–0.280
	Rainbow	157.8–1885.4	101.2–1375.7	0.066–0.212

We emphasize the importance of the key size because, as it was shown in [25], the key size of known quantum-resistant schemes can grow from a few to many kilobytes which can arise some difficulties for today's existing infrastructures of X.509 certificates. A good example is that post-quantum TLS handshake takes 40 KB which is 24 times more expensive [30]. Even worse, there would be some scenarios which are very sensitive to delays that cannot store big certificates or perform signature (generation or verification)

because of those limitations. In such scenarios, most post-quantum signatures would be impractical because of their required computational cost.

Still under study is the Identity-Based Encryption (IBE) scheme which is considered an alternative to traditional certificate-based public key cryptography to reduce communication overheads in wireless sensor networks. In [30], it has been found that ID-based TLS is 2.8× costlier than certificate-based TLS in the pre-quantum scenario.

8. Conclusions

We have detailed the steps to enhance the Lizama's non-invertible key exchange protocol to be used as a public key cryptosystem with single and multiple certification domains. We have provided the specification the certification authority keys and the method to certify the user's public keys. Therefore, our approach is scalable and interoperable and can be exploited in the pre-quantum and the quantum era because the protocol exhibits indistinguishability of the integers in the public key and ciphertexts.

We found that public keys size in Lizama's protocol has the smallest size regarding main post-quantum systems: 0.256 kilobytes and 0.384 kilobytes for public key and certified key, respectively. Moreover, we suggest that the public key database only stores one component of the two integers which are part of the public key, while the second component can be transferred directly to the remote destination. This strategy reduces the required storage space of certified keys to 0.256 kilobytes. Therefore it makes manageable some issues caused by large certificates as fragmentation, segmentation and caching.

Furthermore, we have discussed a method to achieve perfect forward secrecy (PFS) so that a session key can be derived from the previous one and the procedure is repeated as many times as necessary.

Author Contributions: Conceptualization, L.A.L.-P.; Data curation, L.A.L.-P.; Formal analysis, L.A.L.-P.; Investigation, J.M.L.R.; Methodology, J.M.L.R.; Project administration, J.M.L.R.; Resources, J.M.L.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A

Appendix A.1. RSA Cryptosystem

The security of RSA cryptosystem relies on the difficulty of the integer factorization problem. Two invertible numbers e and d are chosen inside the ring defined by $\mathbb{Z}_{\phi(n)}$, so that $e \cdot d \equiv 1 \bmod \phi(n)$. The other ring $\mathbb{Z}_n$ is prepared with $n = p \cdot q$ where p and q are secret prime integers [31]. The encrypted message is computed as $C = M^e \bmod n$ while $M = C^d \bmod n$ returns the original cleartext M. The cryptosystem works because of Euler's theorem since $(M^e \bmod n)^d \bmod n = M^{ed} \bmod n$ but $e \cdot d = k\phi(n) + 1$, so $M^{k\phi(n)+1} \bmod n = M^{k\phi(n)} \cdot M^1 \bmod n$ which yields M provided $M < n$.

Appendix A.2. Diffie–Hellman Key Exchange

Diffie–Hellman key exchange (DH) was the first public key exchange algorithm [20]. The integer prime p defines a ring $\mathbb{Z}_p$ and the generator g is a primitive root module p. The integers p and g are publicly known.

Alice chooses randomly the exponent x_a and she computes $k_a = g^{x_a} \bmod p$ which she sends to Bob over a public channel. On the other side, Bob obtains $k_b = g^{x_b} \bmod p$, then he communicates this integer number to Alice across the channel.

Alice and Bob execute exponentiation over the received number, such that Alice's gets $(g^{x_b} \bmod p)^{x_a} \bmod p = g^{x_b x_a} \bmod p$. Conversely Bob gets $(g^{x_a} \bmod p)^{x_b} \bmod p = g^{x_a x_b} \bmod p$. The two operations yield the same integer number because multiplication of exponents is commutative. The security of the secret shared key relies on the difficulty that given g, k_a and k_b it is computationally infeasible to derive $g^{x_a x_b} \bmod p$.

Appendix A.3. Prefix Attack

Consider the protocol running with $n = 4r$ over a public channel. When an eavesdropper captures the integers from the public channel, where one of them, say w_a, is a prefix of the second number written as $w_{ab} = w_a \cdot k_b \mod 4r$. To derive k_b, the attacker computes the inverse of the prefix that is $(w_a)^{-1}$ to factorize it from the second number. However, in $\mathbb{Z}_{4r}$ w_a and w_{ab} are non-invertible integers, thus the attacker must perform first multiplication by 2^{-2} changing the module from $4r$ to r.

Therefore, if the eavesdropper has captured w_a and w_{ab} from the public channel, they proceed to divide them by 4 thus getting $w_a{}'$ and $w_{ab}{}'$. The eavesdropper computes $(w_a{}')^{-1}$ and they get $(w_a{}')^{-1} \cdot w_{ab}{}'$. As a consequence, the eavesdropper obtains $k_b \mod r$ provided $k_b < r$. To avoid a prefix attack, k_b must be chosen to be greater than the integer prime r. The steps are indicated as follows:

$$
\begin{aligned}
&w_a = 4x_a \cdot k_a \mod 4r \\
&w_{ab} = w_a \cdot k_b = 4x_a \cdot k_a \cdot k_b \mod 4r \\
&w_a{}' = w_a \cdot 4^{-1} = x_a \cdot k_a \mod r \\
&(w_a{}')^{-1} = (x_a \cdot k_a)^{-1} \mod r \\
&k_b = (w_a{}')^{-1} \cdot w_{ab}{}' = k_b \mod r
\end{aligned}
$$

Appendix A.4. Multiplication-Based Attack

Consider again that $p = q = 2$, then $\phi(4r) = 2r - 2$. If the eavesdropper knows P which is computed as $P = 2^{2x}k \bmod 4r$, we affirm that they cannot derive $2^{2x}k \bmod r$ because they ignore $2^{2x}k$. However, after dividing P by 4 they get $2^{2x-2}k \bmod r$. The eavesdropper can perform the product of the public components P and Q:

$$
\begin{aligned}
&P = 2^x k \bmod 4r, \\
&Q = 2^{2r-1-x}k \bmod 4r \text{ because } y = 2r - 2 - x + 1 \\
&P \cdot Q = 2^{2r-1}k^2 \bmod 4r \\
&P \cdot Q \cdot 2^{-2} = 2^{2r-3}k^2 \bmod r \\
&k^2 \equiv P \cdot Q \cdot 2^{-2} \cdot (2^{2r-3})^{-1} \bmod r
\end{aligned}
$$

As a result, the eavesdropper can derive the private key k. To avoid such attack, the exponent is chosen to be $2x$ instead of x:

$$
\begin{aligned}
&P = 2^{2x}k \bmod 4r, \\
&Q = 2^{2r-1-x}k \bmod 4r \text{ where } x < 2r - 1 \\
&P \cdot Q = 2^{x+2r-1}k^2 \equiv 2^x\ 2^{2r-1}k^2 \bmod 4r
\end{aligned}
$$

In this case, the eavesdropper cannot compute the multiplicative inverse of 2^x because they do not know x and they cannot obtain k.

References

1. Shor, P.W. Algorithms for quantum computation: Discrete logarithms and factoring. In Proceedings of the 35th Annual Symposium on on Foundations of Computer Science, Santa Fe, NM, USA, 20–22 November 1994.
2. Barreno, M.A. The Future of Cryptography under Quantum Computers. Dartmouth College Undergraduate Theses, 23 July 2002. Available online: https://digitalcommons.dartmouth.edu/senior_theses/23 (accessed on 11 February 2021)
3. Laboratory, I.T. PQC Standardization Process: Third Round Candidate Announcement. Available online: https://csrc.nist.gov/news/2020/pqc-third-round-candidate-announcement (accessed on 11 February 2021).
4. Chen, L.; Jordan, S.; Liu,Y.-K.; Moody, D.; Peralta,R.; Perlner, R.; Smith-Tone, D. Report on Post-Quantum Cryptography. Available online: http://cm.1-s.es/2017/nistir_8105_draft.pdf (accessed on 11 February 2021).
5. Lizama-Perez, L.A. Non-Invertible Key Exchange Protocol. *SN Appl. Sci.* **2020**, *2*, 1–13. Available online: https://link.springer.com/content/pdf/10.1007/s42452-020-2791-3.pdf (accessed on 11 February 2021). [CrossRef]
6. Grover, L.K. A Fast Quantum Mechanical Algorithm for Database Search. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996.
7. Bennett Ch, H.; Brassard, G. Quantum cryptography: Public key distribution and coin tossing. *arXiv* **2020**, arXiv:2003.06557.
8. Lizama-Pérez, L.A.; López, J.M.; De Carlos-López, E.; Venegas-Andraca, S.E. Quantum flows for secret key distribution in the presence of the photon number splitting attack. *Entropy* **2014**, *16*, 3121–3135. [CrossRef]

9. Lizama-Pérez, L.A.; López, J.M.; De Carlos López, E. Quantum key distribution in the presence of the intercept-resend with faked states attack. *Entropy* **2017**, *19*, 4. [CrossRef]
10. Lizama-Perez, L.A.; López, J.M. Quantum key distillation using binary frames. *Symmetry* **2020**, *12*, 1053. [CrossRef]
11. Bernstein, D.J.; Lange, T. Post-quantum cryptography. *Nature* **2017**, *549*, 188–194. [CrossRef] [PubMed]
12. Wang, S.; Zhu, Y.; Ma, D.; Feng, R. Lattice-based key exchange on small integer solution problem. *Sci. China Inf. Sci.* **2014**, *57*, 1–12. [CrossRef]
13. Mao, S.; Zhang, P.; Wang, H.; Zhang, H.; Wu, W. Cryptanalysis of a lattice based key exchange protocol. *Perspect. Sci.* **2016**, *8*, 228–230. [CrossRef]
14. Jao, D.; and De Feo, L. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *Post-Quantum Cryptography. PQCrypto 2011;* Lecture Notes in Computer Science; Yang, B.Y., Ed.; Springer: Berlin/Heidelberg, Germany, 2011.
15. Costello, C.; Longa, P.; Naehrig, M. Efficient algorithms for supersingular isogeny diffie-hellman. In *Advances in Cryptology—CRYPTO 2016. CRYPTO 2016*; Lecture Notes in Computer Science; Robshaw, M., Katz, J., Eds.; Springer: Berlin/Heidelberg, Germany, 2016.
16. Matsumoto T.; Imai, H. Public quadratic polynomial-tuples for efficient signature-verification and message-encryption. In *Advances in Cryptology—EUROCRYPT '88. EUROCRYPT 1988*; Lecture Notes in Computer Science; Barstow, D., Ed.; Springer: Berlin/Heidelberg, Germany, 1988.
17. Merkle, R.C. Method of Providing Digital Signatures. US Patent 4,309,569, 5 January 1982.
18. Lizama-Perez, L.A. Digital signatures over hash-entangled chains. *SN Appl. Sci.* **2019**, *1*, 1–8.
19. Lizama-Pérez, L.A.; Montiel-Arrieta, L.J.; Hernández-Mendoza, F.S.; Lizama-Servín, L.A.; Eric, S.-A. Public hash signature for mobile network devices. *Ing. Investig. Tecnol.* **2019**, *20*, 1–10. Available online: https://pdfs.semanticscholar.org/fce5/99b5af03457e4b94e123e575d1daca8e24ab.pdf (accessed on 11 February 2021). [CrossRef]
20. Diffie W.; Hellman, M. New directions in cryptography. *IEEE Trans. Inf. Theory* **1976**, *22*, 644–654. [CrossRef]
21. Koblitz, N. Elliptic curve cryptosystems. *Math. Comput.* **1987**, *48*, 203–209.
22. Miller, V.S. Use of elliptic curves in cryptography. In *Advances in Cryptology — CRYPTO '85 Proceedings. CRYPTO 1985. Lecture Notes in Computer Science*; Williams, H.C., Ed.; Springer: Berlin/Heidelberg, Germany, 1986.
23. Bindel, N.; Herath, U.; McKague, M.; Stebila, D. Transitioning to a quantum-resistant public key infrastructure. In *Post-Quantum Cryptography. PQCrypto 2017*; Lecture Notes in Computer Science; Lange, T., Takagi, T., Eds.; Springer: Berlin/Heidelberg, Germany, 2017.
24. Pradel, G.; Mitchell, C.J. Post-quantum certificates for electronic travel documents. In *Computer Security. ESORICS 2020*; Lecture Notes in Computer Science; Boureanu, I., Ed.; Springer: Berlin/Heidelberg, Germany, 2020.
25. Kampanakis, P.; Panburana, P.; Daw, E.; Van Geest, D. The viability of post-quantum X.509 certificates. *IACR Cryptol. ePrint Arch.* **2018**, *2018*, 63.
26. Polk, W.; Housley, R.; Bassham, L. Algorithms and identifiers for the internet X.509 public key infrastructure certificate and certificate revocation list (crl) profile. *Algorithms* **2002**, *2*, 26.
27. Gerck, E. Overview of Certification Systems: X.509, ca, pgp and Skip. Available online: https://www.blackhat.com/presentations/bh-usa-99/EdGerck/certover.pdf (accessed on 11 February 2021).
28. Abdul-Rahman, A. The Pgp Trust Model. *EDI-Forum J. Electron. Commer.* **1997**, *10*, 27–31. Available online: https://ldlus.org/college/WOT/The_PGP_Trust_Model.pdf (accessed on 11 February 2021).
29. NIST Round 3 Finalists. Available online: https://pqc-wiki.fau.edu/w/Special:DatabaseHome (accessed on 11 February 2021).
30. Banerjee, U.; Chandrakasan, A.P. Efficient Post-Quantum TLS Handshakes using Identity-Based Key Exchange from Lattices. In Proceedings of the 2020 IEEE International Conference on Communications (ICC), Dublin, Ireland, 7–11 June 2020. [CrossRef]
31. Rivest, R.L.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126.