cryptography

MDPI

Article

XOR Chain and Perfect Secrecy at the Dawn of the Quantum Era †

Luis Adrián Lizama-Pérez

Departamento de Electrónica, Universidad Técnica Federico Santa María, Av. Vicuña Mackenna 3939, San Joaquín, Santiago 8940897, Chile; luis.lizamap@usm.cl

† To the memory of Ricardo García Pérez.

Abstract: In this article, we present a new method that achieves Shannon's perfect secrecy. To achieve this property, we will introduce the triple XOR cancellation rule. The approach has two execution modes: digital signature and data encryption. We provide perfect secrecy proof of the encryption method. Furthermore, based on our fundamental algorithm, we developed a new strategy for the blockchain system that does not require proof of work (PoW). However, it is a practical mechanism for connecting blocks to the chain. Due to the risk that quantum computers present for current cryptosystems based on prime factorization or discrete logarithm, we postulate that our method represents a promising alternative in the quantum era. We expect our work to have profound implications for the security of communications between mobile devices, the Internet of Things (IoT), and the blockchain.

Keywords: digital signature; encryption; blockchain; XOR function; hash chain; perfect secrecy

check for updates

Citation: Lizama-Pérez, L.A. XOR Chain and Perfect Secrecy at the Dawn of the Quantum Era. *Cryptography* **2023**, *7*, 50. https://doi.org/10.3390/cryptography7040050

Academic Editor: Avinash Srinivasan

Received: 2 August 2023
Revised: 9 October 2023
Accepted: 11 October 2023
Published: 13 October 2023

1. Introduction

The world economy is highly dependent on digital transactions that take place over the internet. Therefore, it is crucial to ensure the authenticity and security of the data by using cryptographic techniques. Public key cryptography is utilized in this scenario to authenticate users and maintain services via digital signatures and the blockchain.

As quantum computing threatens the viability of current algorithms, public key cryptography is undergoing a significant upgrade [1–3]. The quantum algorithm developed by Peter Shor is capable of breaking the security of algorithms based on the computational problem of factoring large prime numbers [4]. Fortunately, hash functions and secret key cryptography are resistant to quantum computing since, at least theoretically, increasing the sizes of the keys is sufficient to prevent the quantum exhaustive search algorithm from being used [5]. The one-time pad (OTP) cryptosystem should be included in the post-quantum category [6]. The encryption formula, which is so simple, relies on the XOR logical operation, as stated by

$$e = m \oplus k \tag{1}$$

where e is the encrypted message, m is the cleartext, and k denotes a random key of the same size as the message to be transmitted. Because there is currently no low-cost, workable method for creating an arbitrarily long key, the OTP encryption scheme, despite its perfect secrecy, is largely theoretical.

1.1. Research Motivation

Our goal is to create a cryptosystem that provides perfect secrecy. The main challenge is in achieving system efficiency. As stated before, perfect secrecy is theoretically possible but not practically feasible because a key size as large as the data size is required. This is where our research problem arises: Is it possible to preserve OTP perfect secrecy while using the triple XOR cancellation? Second, could the system behave efficiently? Furthermore, this research aims to extend the scope of previously known cryptographic techniques in the following dimensions:

1. Scientific: There has been little research on key reusing under OTP. According to [7], a key that is derived from quantum key distribution (QKD) can be reused without risk if an attacker's presence is not discovered during quantum transmission. On the other hand, it has not been demonstrated that OTP perfect secrecy cannot be efficiently achieved. Indeed, chaos systems have been used as pseudo-random number generators (PRNGs), and chaotic cryptography-based systems have been investigated [8,9]. Yet, keys and ciphertext can exhibit short periods, and there is no systematic method for detecting weak keys [10]. It has been demonstrated that OTP is equivalent to finding the initial condition on a pair of binary maps [11]. In this work, we will introduce a cryptosystem that we call 3-encryption, which uses the rule of triple cancellation to achieve perfect secrecy.
2. Technological: Because it is well-known that current public key data protection mechanisms do not resist quantum cryptanalysis, the development of new cryptographic security schemes must be prioritized. Despite the fact that the National Institute of Standards and Technology (NIST) has published a set of post-quantum algorithms, the security evaluations and discussions of such algorithms (regarding potential vulnerabilities) are ongoing. These algorithms are used in encryption, signing, and key establishment. Nevertheless, our basic scheme can be simultaneously used for block chaining, data encryption, and digital signatures. As a result, we envision a multifunctional cryptographic platform that is capable of providing integrated security services.
3. Security: We will demonstrate that the encryption scheme is capable of achieving perfect secrecy. Surprisingly, our algorithm's security properties are evaluated using XOR, hash, and integer addition, which are simple to analyze, and require no complex mathematical formalism. In the appendix of this document, we will show how the encryption method achieves perfect secrecy.

1.2. State of the Art

Given the imminent development of quantum computers, the replacement of current public key cryptography methods based on integer factorization and the discrete logarithm cannot be postponed. There are various approaches to implement post-quantum cryptography: lattice-based, hash-based, code-based, and multivariate polynomial cryptography are the most significant types.

- Lattice-based cryptography is based on the shortest vector problem's hardness, which is proven to be difficult to solve with quantum computers. In the N-th-degree truncated polynomial ring units (NTRUs), lattice-based encryption parameters must be carefully chosen to prevent known attacks [12,13].
- The security of hash-based cryptography is inextricably linked to the hash function that is used, such as pre-image resistance and collision resistance. The signature scheme with hash-based instantiation of a narrow collision resistance function (SPHINCS+) has been chosen by NIST [14,15].
- Error-correcting codes are the foundation of code-based cryptography, functioning well for public key encryption. To ensure security, nearly all implemented algorithms in this class employ large keys. The McEliece public key encryption system and the Niederreiter cryptosystem are the most representative examples of code-based cryptography [16,17].
- The difficulty of solving systems of multivariate equations over finite fields underpins multivariate polynomial cryptography. Rainbow could serve as the foundation for a quantum secure digital signature because it produces short signatures [18,19].
- Isogeny-based cryptography relies on the difficulty of finding a certain mapping (called isogeny) between two given supersingular elliptic curves. The Diffie–Hellman and elliptic curve Diffie–Hellman key-exchange methods can be replaced with the

supersingular isogeny Diffie–Hellman key exchange (SIKE) as a quantum-resistant alternative [20].

- Symmetric key cryptosystems are resistant to quantum computer attacks as long as they use sufficiently large key sizes [21,22]. According to [23], in this study, we suggest 256-bit keys.

In 2022, the NIST announced four candidates to be standardized, plus fourth-round candidates. One of the chosen methods is a public key encapsulation technique (KEM): CRYSTALS-Kyber (lattice-based), and three digital signature algorithms: CRYSTALS-Dilithium (lattice-based), FALCON (lattice-based), and SPHINCS+(hash-based) [24–27]. BIKE (code-based), HQC (code-based), and SIKE (elliptic-curves-based) advance to the fourth round [24]. Unfortunately, the SIKE algorithm, one of these four candidates, has been broken at NIST's security level 1 [28]. Even worse, the rainbow post-quantum signature scheme (multivariate quadratic equations type) has been revealed to be broken [29,30].

For this study, we classify digital signature schemes into those that verify the signature through a mathematical trapdoor function, such as Rivest–Shamir–Adleman (RSA), the digital signature algorithm (DSA), and NIST post-quantum standards [31–33]. In the second approach, we consider those that perform signature verification by means of a hash chain. The Merkle tree-based signature [34], SPHINCS+ [15], the hash chain protocol [35], and the HMAC-based digital signature [36] are all grouped into this category. Such systems are post-quantum methods because they are based on the security properties of hash functions. The signature method we will introduce here uses block chaining; however, the verification is done with the XOR function and just one hash operation.

We developed the XOR digital signature as the first scheme under this approach. We have, however, expanded this scheme to perform data encryption. The resulting cipher scheme is similar to the block chaining of today's most widely used encryption algorithms, such as the advanced encryption standard (AES), and Blowfish [21,37]. So, we will use the encrypted chaining mode (CBC) as the reference system [38,39]. In contrast to these methods, our approach does not require an iterative process of substitution and permutations, implying spacetime complexity. Furthermore, the cryptosystem achieves perfect secrecy, which none of the other schemes do.

Currently, Currently, blockchain is a widely disruptive technology that enables the permanent recording of transactions across interconnected computer systems [40,41]. The most serious threat to blockchain is the imminent appearance of quantum computers capable of breaching its security [42]. The power consumption required during the process of placing blocks on the chain is the second most serious concern. While there are other schemes that address energy consumption, such as Ethereum [43] and Litecoin [44,45], none of them constitute a post-quantum system. Post-quantum blockchain is explained in [46], whereas quantum and hybrid quantum/classical blockchain protocols are discussed in [47].

A quantum one-time pad (QOTP) is similar to the traditional one-time pad approach. The main distinction is that an application of the X and Z Pauli gates constitutes the quantum operation instead of the XOR operation between the cleartext and key [48]. Furthermore, quantum properties of matter can be used to define a peer-to-peer quantum cash network, which may be realized in the future quantum internet [49]. The scheme, called qBitcoin, incorporates a method to perform quantum digital signatures [50].

1.3. Perfect Secrecy

According to Claude Shannon's basis of secret communication, it is well-known that OTP allows perfect secrecy [51], at least in theory, as there are currently no real-world cryptosystems that only use the XOR encryption function (see Figure 1). Even if the attacker has unlimited computational capacity, an encrypted text has perfect secrecy if the attacker's knowledge of the message's content is the same before and after the adversary inspects the encrypted text. This means that the encrypted message gives the adversary no precise information about the cleartext message's content. This is due to the fact that

OTP requires that the encryption key's size match that of the cleartext message [51,52] Unfortunately, establishing an arbitrarily large secret key between Alice and Bob—the parties seeking private communication—presents a significant challenge.

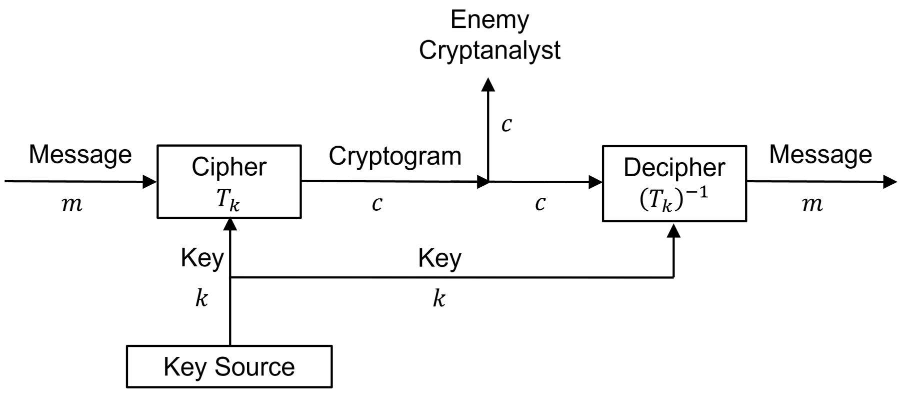


Figure 1. The general scheme of the Shannon encryption scheme.

Let us enumerate the properties of perfect secrecy [53,54]:

- The number of possible keys is greater than or equal to the number of possible plaintexts.
- The key is selected uniformly at random from the key space.
- A key should only be used once.

There are other cryptosystems besides OTP that can provide perfect secrecy. For example, the Vignère Cipher is a cryptosystem with perfect secrecy, as long as the following conditions are met: (1) the keyword's length matches that of the plaintext; (2) it is a randomly generated string of letters; and (3) the keyword is only used once [53].

What is important to emphasize here is that under perfect secrecy, an exhaustive key search is pointless because every possible plaintext is a valid candidate. Whatever the complexity of the cryptosystem, the attacker can only guess the plaintext. The novelty of our work is that we designed our cryptosystem mainly with the XOR encryption/decryption relations $e = m \oplus k$ and $m = e \oplus k$. Since $k \oplus k = 0$, the decryption relation is valid because the XOR operation is the only reversible Boolean operation; thus, $m = e \oplus k = m \oplus k \oplus k$, where k represents the secret key and m denotes the plaintext.

This article will demonstrate that it is not strictly necessary to pre-establish the entire secret key beforehand in order to obtain perfect secrecy. It should be noted that using a key re-establishment scheme is another simple option, but it is impractical because it requires the ongoing exchange between a new secret key and remote users for each round, which can be written as

$$m_i \oplus k_i = e_i \text{ for } i = 1, 2, \ldots \tag{2}$$

where i denotes the iteration round. We will return to this point later when we discuss XOR encryption.

1.4. Triple XOR Cancellation Rule

Let us say that k_0, k_1, and k_2 represent three binary strings. The following operations are used to derive z_{01}, z_{02}, and z_{12}: $z_{01} = k_0 \oplus k_1$, $z_{02} = k_0 \oplus k_2$, and $z_{12} = k_1 \oplus k_2$. The additive group $\mathbb{Z}_\oplus$ is defined by z_{01}, z_{02}, and z_{12} because the following operations can be directly verified as

$$\begin{aligned} z_{01} \oplus z_{02} &= z_{12} \\ z_{02} \oplus z_{12} &= z_{01} \\ z_{01} \oplus z_{12} &= z_{02} \end{aligned} \tag{3}$$

Let ${k_0}'$ be a random variable, then compute ${k_1}'$, such that ${k_0}' \oplus {k_1}' = z_{01}$. Now, ${k_2}'$ can be chosen, such that ${k_0}' \oplus {k_2}' = z_{02}$. Thus, we want to find out if ${k_1}' \oplus {k_2}' = z_{12}$. To establish it, we know that z_{01} and z_{02} imply the existence of z_{12} due to the closure of the additive group. Since ${k_0}'$, ${k_1}'$ and ${k_2}'$ define the same group $\mathbb{Z}_{\oplus}$, we have ${k_1}' \oplus {k_2}' = z_{12}$. Since ${k_0}'$ was defined randomly, this implies that for every ${k_0}'$, there exist ${k_1}'$ and ${k_2}'$, which define the same group $\mathbb{Z}_{\oplus}$. Suppose Alice has k_0, k_1, and k_2, then she computes z_{01}, z_{02}, and z_{12}. If Alice keeps hidden k_0, k_1, and k_2 but publicly shares z_{01}, z_{02}, and z_{12}, an adversary can derive another set of numbers, ${k_0}'$, ${k_1}'$, and ${k_2}'$, which return z_{01}, z_{02}, and z_{12} as well. However, due to perfect secrecy in XOR, Eve cannot derive the original Alice set, k_0, k_1, and k_2. As a result, we are able to identify Alice as having an advantage, which allows us to arrive at our fundamental security rule to transmit a message m_0 from Alice to Bob, which is illustrated graphically in Figure 2.

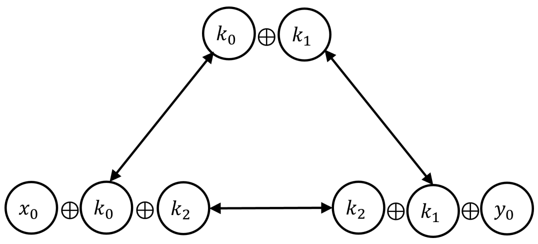


Figure 2. Triple XOR cancellation rule. Here, k_0, k_1, and k_2 are the initial shared secret keys. The number x_0 is a random number chosen by Alice and y_0 is computed as $y_0 = h_0 \oplus x_0$, where $h_0 = f(m_0)$, and f is the hash function applied to the message m_0.

1.5. Proposal of Our Approach

In earlier works, we introduced hash functions and HMAC-based digital signature techniques [36,55]. This time, we will base our algorithms on the characteristics of the XOR function. We claim that the following cryptographic primitives can be supported using our method.

- Digital signature: It is impossible for the adversary to send a message pretending to be from her because each user has a public key in their name. By using the XOR signature algorithm, users can make sure that the current message is connected to every previous message in the chain, starting with the user identification message. This topic will be discussed in Section 2.
- XOR chain: The XOR signature model has allowed us to define a new approach to the blockchain system that we call the XOR chain. We will first present a hash function-based game called Crypto Bingo in order to conceptualize the XOR chain. Section 3 covers this subject.
- Data encryption: The messages are encrypted so that it is impossible for the attacker to see their content; however, authorized recipients can recover the messages in original plain text. Section 4 of the document will address XOR encryption. In Appendix A, we show the perfect secrecy demonstration.

The introduced schemes are compared to related works in Section 5, which also discusses some implementation details that highlight the main opportunities and challenges to take into account. The findings of this research work serve to conclude the document.

2. Digital Signature

In a digital signature, the holder of the public key and the signed message are inextricably linked. A digital signature makes it possible to operate systems remotely, like payment systems, and it ensures mobile device authentication, signed software downloads, and certified web browsing. In this section, we will introduce our digital XOR signature scheme. Nevertheless, the scheme is based on signature techniques that we previously developed. Therefore, Section 2.1 explains the hash function-based blockchain scheme.

Section 2.2 describes the method using HMAC functions. In Section 2.3, we introduce the digital signature method that uses the XOR function.

2.1. The Hash Chain Protocol

This protocol is based on the mathematical characteristics of a lengthy hash chain. For Alice to send Bob a signed message, users must complete the following actions [35]:

To create a pair of keys (public and private), Alice chooses a random number that we call the seed s_a. She computes the hash chain as $f^{l_n}(s_a)$, where f denotes the hash function of the cryptosystem. The hash chain's length (the number of times the hash function is applied to the seed s_a) is indicated by the exponent l_n. The public key becomes $f^{l_n}(s_a)$.

On the other hand, Alice's secret value s_a is used to define the private key $f^{l_n-i}(s_a)$, where $1 \leq i \leq l_n$. This implies that $f^{l_n-1}(s_a), f^{l_n-2}(s_a), \dots f(s_a)$ constitute the set of Alice's private keys, as represented in Table 1. Let us write the hash of the message to be signed as $f(m) = h$. The protocol is summarized below:

1. Alice and Bob generate their hash chain, which allows them to define their public and private keys.
2. They share their public keys over the public channel.
3. Alice computes $f^{l_n-h}(s_a)$ and publishes it along m.
4. Bob—or any user who wants to verify the signature—just computes h, then $f^h(f^{l_n-h}(s_a)) == f^{l_n}(s_a)$.
5. Alice shares a new public key.

Table 1. The public and private keys of the hash chain protocol are specified, where $i(j) \geq 1$.

User	Public Key	Private Keys
Alice	$f^{l_n}(s_a)$	$f^{l_n-i}(s_a)$
Bob	$f^{l_n}(s_b)$	$f^{l_n-j}(s_b)$

Providing the eavesdropper uses the hash values on the right side of the signature hash, and assuming we can generate chains that long (a detailed discussion can be found in [35], the protocol is vulnerable to a man-in-the-middle (MITM) attack. For example, the hash chain at the top of Figure 3 allows Bob to verify the signature of m_0, taking Alice's public key $f^{l_n-h_0}(s_0)$ and, thus, checking that $f^{h_0}(f^{l_n-h_0}(s_0)) == f^{l_n}(s_0)$. After Alice publishes her new public key $f^{l_n}(s_1)$ (see the hash chain at the bottom of Figure 3), she signs m_1, publishing $f^{l_n-h_1}(s_1)$. However, the attacker can use $f^{l_n-h'_1}(s_1)$ to fake it because she can compute all the hash values at the right of the signature hash.

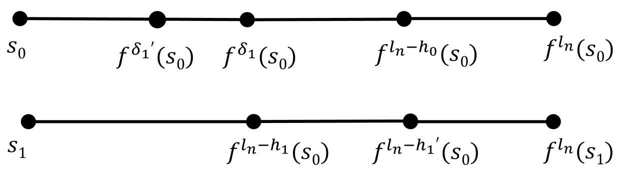


Figure 3. The hash values of the hash chain protocol.

The entanglement algorithm, which is based on the following facts, was proposed to overcome this weakness:

- An attacker can only exploit the hash values at the right-hand side of the signature hash in the current hash chain, which implies that $h'_i < h_i$.
- After a message is signed, some hash points to the left of the signature hash always remain unused in the previous hash chain.

If h'_i is the hash value of the (fake) message that the attacker is trying to pass off as authentic, then as a result of Equation (4), Eve cannot compute $f^{\delta_1}(s_0)$ because $\delta'_1 < \delta_1$, which implies that $f^{\delta'_1}(s_0)$ is at the left of $f^{\delta_1}(s_0)$, as illustrated in Figure 3.

$$\begin{aligned} h'_1 &< h_1 \\ \frac{h'_1}{l_n} &< \frac{h_1}{l_n} \\ (l_n - h_0)\frac{h'_1}{l_n} &< (l_n - h_0)\frac{h_1}{l_n} \\ h'_1 - \frac{h_0 h'_1}{l_n} &< h_1 - \frac{h_0 h_1}{l_n} \end{aligned} \tag{4}$$

If we define $\delta_1 = h_1 - \frac{h_0 h_1}{l_n}$, then $f^{\delta'_1}(s_0) < f^{\delta_1}(s_0)$. Thus, to authenticate $f^{\delta'_1}(s_0)$, Equation (5) is used.

$$f^{l_n - h_0 - \delta_1}(f^{\delta_1}(s_0)) = f^{l_n - h_0}(s_0) \tag{5}$$

Let us describe the verification steps of the protocol:

1. Given that Alice has previously published $f^{l_n}(s_1)$, to sign the message m_1, Alice computes δ_1 to obtain $f^{\delta_1}(s_0)$, then she publishes it along m_1.
2. Using m_0 and m_1, Bob computes δ_1 and verifies two conditions:
   (i) $f^{l_n - h_0 - \delta_1}(f^{\delta_1}(s_0)) = f^{l_n - h_0}(s_0)$.
   (ii) $f^{h_1}(f^{l_n - h_1}(s_1)) = f^{l_n}(s_1)$.

2.2. Digital Signatures Based on HMAC

Digital signatures can be created by taking advantage of the hash-based message authentication code (HMAC). Alice will carry out the steps listed below if she wants to send Bob a signed message [36,56], where the symbol $< >_k$ denotes the HMAC function applied with key k:

1. $A \rightarrow B : < m >_{f^{l_n - i}(s_a)}$. Alice signs the message (m) by applying the HMAC function and using her private key $f^{l_n - i}(s_a)$.
2. $A \leftarrow B :\ f^{l_n - j}(s_b)$. Bob sends his private key $f^{l_n - j}(s_b)$ to Alice.
3. $A \rightarrow B :\ m, f^{l_n - j}$. Alice verifies Bob's authenticator because she computes $f^j(f^{l_n - j}(s_b))$, which returns Bob's public key $f^{l_n}(s_b)$. Then she sends $f^{l_n - i}(s_a)$ and the message (m) to Bob.
4. $A \rightarrow B :\ m, f^{l_n - i}$. Bob verifies Alice's authenticator $f^i(f^{l_n - i}(s_a)) == f^{l_n}(s_a)$

Because Alice and Bob are unable to verify the freshness of the authentication keys, $f^{l_n - i}(s_a)$ and $f^{l_n - j}(s_b)$, the protocol is susceptible to replay attacks because an eavesdropper can capture previously used keys. In [35], a trusted third party that acts as an intermediary between users is introduced to the protocol as a way to move around this restriction. However, the fundamental protocol is still functional for dedicated point-to-point links with synchronized users. In this case, $i = j$ and each new key can be verified from the previous one, i.e., $f(f^{l_n - i}) = f^{l_n - i + 1}$.

The HMAC Signature Method

Let us describe the basic idea of the hash chain-based signature. For this explanation, consider Equation (6)

$$< h_1, f^{\delta_1}(s_0) >_{f^{l_n - h_1}(s_1)} \tag{6}$$

where $f^{l_n - h_1}(s_1)$ is the key of the HMAC function. To validate the signature of the message m_1, Alice sends to Bob m_1, δ_1, the computed HMAC, and the signature hash $f^{l_n - h_1}(s_1)$. In addition, Bob is able to obtain the signature hash $f^{l_n - h_0}(s_0)$ (from a public database). Thus, to validate the signature of m_1, the two relations must be satisfied: the security condition $f^{l_n - h_0 - \delta_1}(f^{\delta_1}(s_0)) = f^{l_n - h_0}(s_0)$ and the signature condition $f^{h_1}(f^{l_n - h_1}(s_1)) = f^{l_n}(s_1)$. The basic protocol previously discussed must be expanded to handle the size of the exponent (the message's hash value), which is accomplished by utilizing multiple chains. To handle 256-bit hashes, 16 chains of length 2^{16} must be used [36].

2.3. Digital XOR Signature

In this scheme, Alice's public and private keys—to perform the XOR signature—are written in Table 2, where k_0, k_1, and k_2 are random binary strings. In this protocol, h_j represents the hash code of the message m_j (the message to be signed). The message m_0 is assumed to be the user's credential containing identification data. The XOR signature process is represented in Figure 4, the execution of variables is illustrated in Table 3, and Figure 5 depicts the message exchange protocol.

Table 2. Alice private/public key definition to perform XOR signatures.

Private Key	**Public Key**
$\{k_0, k_1, k_2\}$	$\{x_0 \oplus k_0 \oplus k_2\}$, $\{y_0 \oplus k_1 \oplus k_2\}$, $\{k_0 \oplus k_1\}$

Table 3. Protocol execution for rounds $0 \ldots j$. The terms c_j, l_j, r_j, and d_j are shown after each execution round. In the last column, we show the results of $c_j \oplus l_j \oplus r_j \oplus d_j$. The authentication rule is $f(m_j) == h_j$, where $h_j = h_{j-1} \oplus c_j \oplus l_j \oplus r_j \oplus d_j$.

Round	$\boldsymbol{m_j}$	$\boldsymbol{c_j = y_{j-1} \oplus k_j \oplus k_{j+1}}$	$\boldsymbol{l_j = k_j \oplus k_{j+2}}$	$\boldsymbol{r_j = y_j \oplus k_{j+1} \oplus k_{j+2}}$	$\boldsymbol{d_j = x_{j-1} \oplus x_j}$	$\boldsymbol{c_j \oplus l_j \oplus r_j \oplus d_j}$
0	m_0	$k_0 \oplus k_1$	$x_0 \oplus k_0 \oplus k_2$	$y_0 \oplus k_1 \oplus k_2$	—	h_0
1	m_1	$y_0 \oplus k_1 \oplus k_2$	$k_1 \oplus k_3$	$y_1 \oplus k_2 \oplus k_3$	$x_0 \oplus x_1$	$h_0 \oplus h_1$
2	m_2	$y_1 \oplus k_2 \oplus k_3$	$k_2 \oplus k_4$	$y_2 \oplus k_3 \oplus k_4$	$x_1 \oplus x_2$	$h_1 \oplus h_2$
3	m_3	$y_2 \oplus k_3 \oplus k_4$	$k_3 \oplus k_5$	$y_3 \oplus k_4 \oplus k_5$	$x_2 \oplus x_3$	$h_2 \oplus h_3$
4	m_4	$y_3 \oplus k_4 \oplus k_5$	$k_4 \oplus k_6$	$y_4 \oplus k_5 \oplus k_6$	$x_3 \oplus x_4$	$h_3 \oplus h_4$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
j	m_j	$y_{j-1} \oplus k_j \oplus k_{j+1}$	$k_j \oplus k_{j+2}$	$y_j \oplus k_{j+1} \oplus k_{j+2}$	$x_{j-1} \oplus x_j$	$h_{j-1} \oplus h_j$

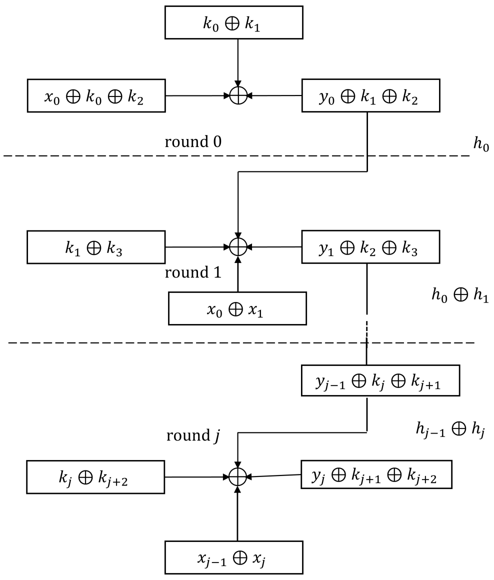


Figure 4. Digital XOR signature process between Alice and Bob.

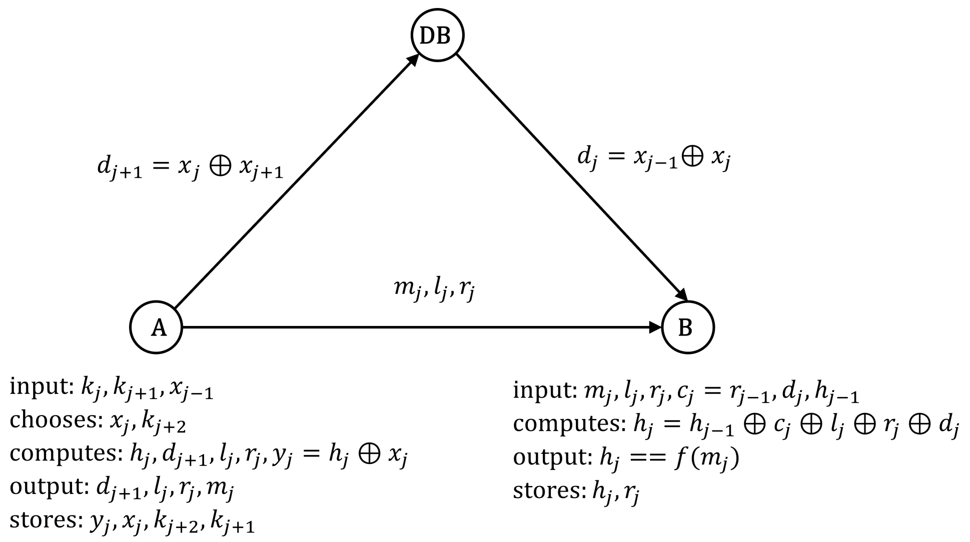

DB
$d_{j+1} = x_j \oplus x_{j+1}$
$d_j = x_{j-1} \oplus x_j$
m_j, l_j, r_j
A
B
input: k_j, k_{j+1}, x_{j-1}
chooses: x_j, k_{j+2}
computes: $h_j, d_{j+1}, l_j, r_j, y_j = h_j \oplus x_j$
output: d_{j+1}, l_j, r_j, m_j
stores: $y_j, x_j, k_{j+2}, k_{j+1}$
input: $m_j, l_j, r_j, c_j = r_{j-1}, d_j, h_{j-1}$
computes: $h_j = h_{j-1} \oplus c_j \oplus l_j \oplus r_j \oplus d_j$
output: $h_j == f(m_j)$
stores: h_j, r_j

Figure 5. Message exchange in round j of the XOR signature algorithm.

From Figure 4, we denote the terms as the center term c_j (the term arriving from up to down to the XOR operator), the down term d_j (the term arriving from down to up to the XOR operator), the left term l_j (the term arriving from left to right to the XOR operator), and the right term r_j (the term arriving from right to left to the XOR operator). So, we can state Equation (7) as follows:

$$\begin{aligned} c_j &= y_{j-1} \oplus k_j \oplus k_{j+1} \\ l_j &= k_j \oplus k_{j+2} \\ r_j &= y_j \oplus k_{j+1} \oplus k_{j+2} \\ d_j &= x_{j-1} \oplus x_j \end{aligned} \tag{7}$$

where $c_j = r_{j-1}$. The equation holds for $j \geq 1$ but the initial round ($j = 0$) can be seen in Figure 4.

In round j, Alice wants to sign m_j. Then she computes $f(m_j) = h_j$ and y_j because $y_j = h_j \oplus x_j$, where x_j is a chosen random number. In addition, Alice computes c_j, l_j, r_j as indicated in Table 3. Then, Alice chooses x_{j+1} to compute $d_{j+1} = x_j \oplus x_{j+1}$ and sends d_{j+1} to DB but retains y_j and x_{j+1} for the next round, as illustrated in Figure 5.

On his side, Bob retrieves d_j from Alice's DB. He computes $f(m_j)$ and h_j as stated by Equation (8), where $c_j = r_{j-1}$. To accept the signature, Bob verifies that $f(m_j) == h_j$. Finally, he keeps h_j and r_j for the next round.

$$h_j = h_{j-1} \oplus c_j \oplus l_j \oplus r_j \oplus d_j \tag{8}$$

2.3.1. Security Analysis

Equation (8) can be rewritten as $h_j^{ba} = h_{j-1}^{\ b} \oplus h_{j-1}^{\ a} \oplus h_j^{\ a}$ (where a denotes Alice, b denotes Bob, and ba denotes the resulting hash code) because $h_{j-1}^{\ a} \oplus h_j^{\ a} = c_j^{\ a} \oplus l_j^{\ a} \oplus r_j^{\ a} \oplus d_j^{\ a}$, where $\{c_j^{\ a}, c_j^{\ a}, c_j^{\ a}, c_j^{\ a}\}$ are the channel terms. However, the term $h_{j-1}^{\ b}$ in the equation is stored on Bob's side, so the eavesdropper has no way to modify it. If Eve mounts a man-in-the-middle (MITM) attack, she inserts her numbers in the channel, say $h_{j-1}^{\ e}, h_j^{\ e}$ (where e denotes Eve), then Equation (8) is $h_j^{be} = h_{j-1}^{\ b} \oplus h_{j-1}^{\ e} \oplus h_j^{\ e}$, which, in turn, results in $f(m_j^{\ e}) \neq h_j^{ba}$, where $m_j^{\ e}$ is Eve's message.

2.3.2. Key Renewal

Alice can act as a server between other users; thus, she could maintain a separate chain with each user. Therefore, Alice executes the following procedure: she keeps the keys k_0 and k_1 but updates k_2, which is achieved by randomly choosing another binary string. As a result, the key $\{x_0 \oplus k_0 \oplus k_1\}$ can be kept unchanged but k_2 in the other two keys $\{y_0 \oplus k_0 \oplus k_2\}$, $\{k_1 \oplus k_2\}$ must be substituted using k_2' and k_2'' for the first two cases. The process is as follows:

- $\{x_0 \oplus k_0 \oplus k_1\}, \{y_0 \oplus k_0 \oplus k_2\}, \{k_1 \oplus k_2\}$
- $\{x_0 \oplus k_0 \oplus k_1\}, \{y_0 \oplus k_0 \oplus k_2'\}, \{k_1 \oplus k_2'\}$
- $\{x_0 \oplus k_0 \oplus k_1\}, \{y_0 \oplus k_0 \oplus k_2''\}, \{k_1 \oplus k_2''\}$
- $\vdots$

This is feasible and safe because, as stated before, an adversary cannot derive k_0, k_1, or k_2, despite knowing $\{x_0 \oplus k_0 \oplus k_1\}$, $\{y_0 \oplus k_0 \oplus k_2\}$, and $\{k_1 \oplus k_2\}$. Any user who wants to authenticate Alice only needs to verify that the key $x_0 \oplus k_0 \oplus k_1$ remains unchanged. Alice maintains a separate chain with each user of the system.

3. Blockchain

Blockchain [40] is an unalterable record that maps the transactions of a distributed service over the internet. It constitutes the support of the transactions of the cryptocurrency system, such as Bitcoin, at a critical moment, in which advances in classical and quantum algorithms are becoming evident [57]. The biggest security concern around blockchain is that it is based on ECDSA, which is vulnerable to Shor's quantum algorithm [4].

Blockchain allows miners to record and link digital transactions in the system. The process is carried out by calculating the hash of the previous transaction's concatenation, a unique random number (nonce), and the root of the Merkle tree, which groups the miner's transactions, as illustrated in Figure 6. If the obtained hash meets a predetermined number of null bits, for example, a prefix of twenty zeros, the miner can add the block to the chain. The correctness of the nonce is efficiently verifiable by all miners on the network. Although blockchain incorporates a method that is suitable for choosing the transaction block without controversy, miners spend large amounts of (electrical) energy in the relentless search for the nonce. In fact, miners need to compute a large number of hashes per unit of time, which entails a large energy cost. In this section, we will discuss a method for choosing a block of transactions that is both verifiable and incontrovertible.

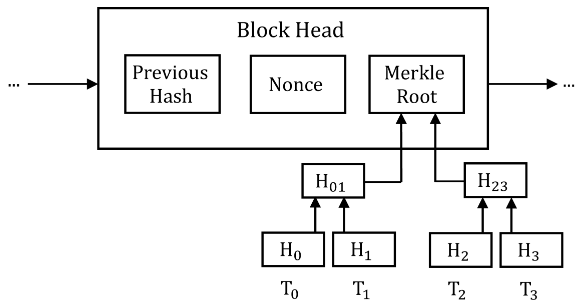


Figure 6. The miners calculate the hash of the concatenation resulting from the previous transaction, a number used only once (nonce), and the root of the Merkle tree, which groups the miner's transactions.

The XOR signature method we introduced in Section 2.3 can be exploited to define a system of transaction-linked blocks to maintain a secure database of such network transactions. To avoid the computational efforts and the required electrical power consumption, the random selection of the miner could be proposed. However, doubts could prevail about the integrity of the random selection algorithm. We will reformulate this scenario through the XOR chain algorithm. However, before introducing the XOR chain, in Section 3.1, we will discuss a similar scheme from which we have developed the XOR chain algorithm, denoted as Crypto Bingo.

3.1. Crypto Bingo

In this section, we will introduce Crypto Bingo, a novel blockchain system that we created in the context of a gaming scenario. The purpose of Crypto Bingo is to clearly and irrefutably determine a winner. The winner will be granted the right to have their list of transactions recorded on the blockchain.

Let us start by stating that each player, say i, registers into the game by posting the binary number z_i that each user chooses randomly. For round j, each player i computes $g_{ij} = f^{x_{ij}}(w_{ij})||t_{ij}$, where $w_{ij} = z_i||h_{ij}{}'$, and $h_{ij}{}'$ is the hash code that belongs to the winning player in the previous round. In addition, x_{ij} is a random number chosen by each node and t_{ij} is the hash code of their Merkle tree, which identifies the transaction group of player i. Let us go deeper into this algorithm (see Figures 7 and 8).

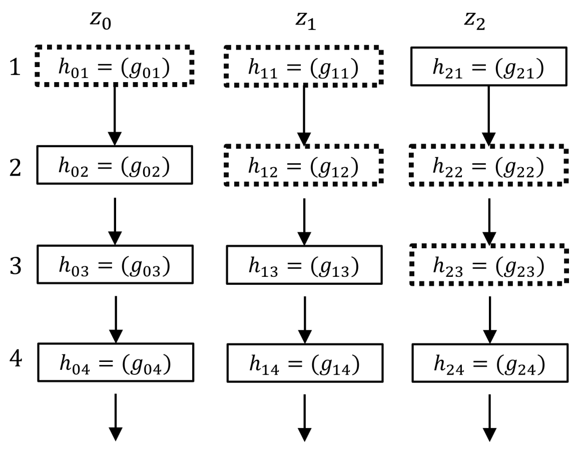


Figure 7. Players register $h_{ij} = f(g_{ij})$, then they announce $g_{ij} = f^{x_{ij}}(w_{ij})||t_{ij}$ and the pair $\{x_{ij}, t_{ij}\}$.

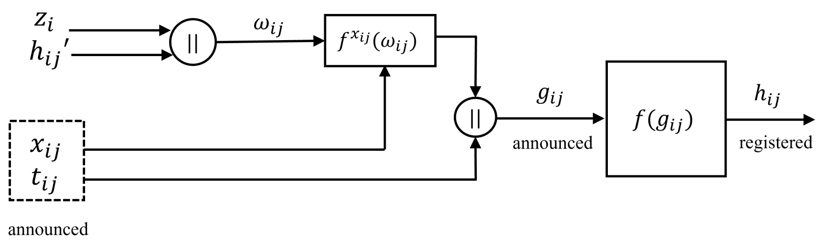


Figure 8. Diagram of the process to compute h_{ij}. Input h'_{ij} represents the hash code of the winner in the previous round.

- In the first round, the root player computes and registers $h_{01} = f(g_{01})$ into the game, where $g_{01} = f^{x_{01}}(w_{01})||t_{01}$ and $w_{01} = z_0$ (instead of $z_0||h_{01}{}'$ because there is no a previous winner). In addition, players compute and register h_{i1} into the game. Then the root player announces g_{01} and the player whose g_{i1} failed the fewest number of bits wins. This value corresponds to the minimum Hamming distance, denoted as δ. It is clear that the probability of obtaining the correct bits (e.g., 256 bits) is quite low. The winner is allowed to place his block into the chain. Let z_1 be the winning player with h_{11}. To be verified, the root player publishes (x_{01}, t_{01}), while player 1 announces (x_{11}, t_{11}). Then, all players verify that they correspond to h_{01} and h_{11}, respectively.

As can be deduced from the previous discussion, verifying the winning player's given h_{ij} and (x_{ij}, t_{ij}) can be performed efficiently. The players do not require high computational efforts and there is no concern about energy consumption. More importantly, none of the players can violate the game's rules because everyone reveals their registration number z_i from the beginning of the game.

3.2. XOR Chain

The XOR chain algorithm is based on the XOR signature model discussed previously, but now we use Equation (9)

$$\begin{aligned} c_j &= y_{j-1} \oplus k_j \oplus k_{j+1} \\ l_j &= k_j \oplus k_{j+2} \oplus x_{j-1} \oplus x_j \\ r_j &= y_j \oplus k_{j+1} \oplus k_{j+2} \end{aligned} \tag{9}$$

where $c_j = r_{j-1}$. The complete execution of the protocol is illustrated in Table 4. In the following, we will refer to Alice as the root player, denoted with z_0 (see Figure 9).

Table 4. We demonstrate the terms c_j, l_j, and r_j after each execution round. The last column shows the result of $c_j \oplus l_j \oplus r_j$. The XOR chain rule is $l_j \oplus r_j == h_j \oplus h_{j+1}$.

Round	$c_j = y_{j-1} \oplus k_j \oplus k_{j+1}$	$l_j = k_j \oplus k_{j+2} \oplus x_{j-1} \oplus x_j$	$r_j = y_j \oplus k_{j+1} \oplus k_{j+2}$	$c_j \oplus l_j \oplus r_j$
0	$k_0 \oplus k_1$	$x_0 \oplus k_0 \oplus k_2$	$y_0 \oplus k_1 \oplus k_2$	h_0
1	$y_0 \oplus k_1 \oplus k_2$	$k_1 \oplus k_3 \oplus x_0 \oplus x_1$	$y_1 \oplus k_2 \oplus k_3$	$h_0 \oplus h_1$
2	$y_1 \oplus k_2 \oplus k_3$	$k_2 \oplus k_4 \oplus x_1 \oplus x_2$	$y_2 \oplus k_3 \oplus k_4$	$h_1 \oplus h_2$
3	$y_2 \oplus k_3 \oplus k_4$	$k_3 \oplus k_5 \oplus x_2 \oplus x_3$	$y_3 \oplus k_4 \oplus k_5$	$h_2 \oplus h_3$
4	$y_3 \oplus k_4 \oplus k_5$	$k_4 \oplus k_6 \oplus x_3 \oplus x_4$	$y_4 \oplus k_5 \oplus k_6$	$h_3 \oplus h_4$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
j	$y_{j-1} \oplus k_j \oplus k_{j+1}$	$k_j \oplus k_{j+2} \oplus x_{j-1} \oplus x_j$	$y_j \oplus k_{j+1} \oplus k_{j+2}$	$h_{j-1} \oplus h_j$

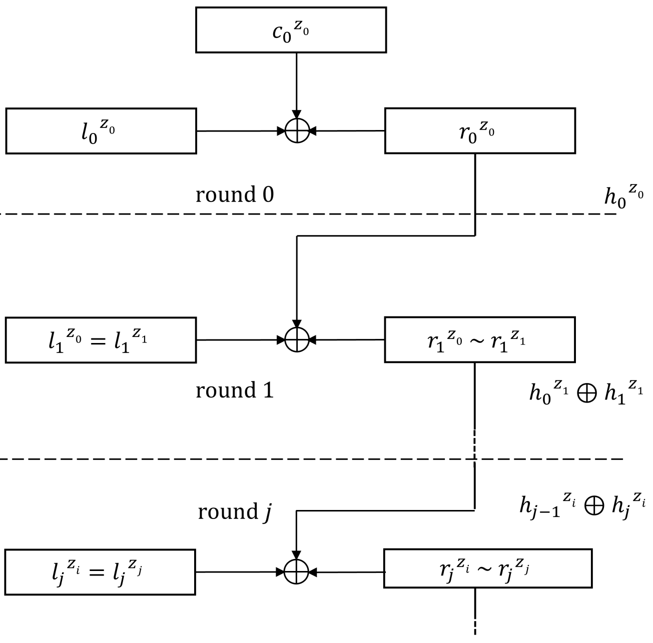

$c_0{}^{z_0}$
$l_0{}^{z_0}$
$r_0{}^{z_0}$
round 0
$h_0{}^{z_0}$
$l_1{}^{z_0} = l_1{}^{z_1}$
$r_1{}^{z_0} \sim r_1{}^{z_1}$
round 1
$h_0{}^{z_1} \oplus h_1{}^{z_1}$
round j
$h_{j-1}{}^{z_i} \oplus h_j{}^{z_i}$
$l_j{}^{z_i} = l_j{}^{z_j}$
$r_j{}^{z_i} \sim r_j{}^{z_j}$

Figure 9. In round 1, z_1 publishes $\{h_0{}^{z_1}, h_1{}^{z_1}\}$ in DB. He computes $l_1{}^{z_1}$ and $r_1{}^{z_1}$ but chooses $k_3{}^{z_1}$, so that $l_1{}^{z_1} = l_1{}^{z_0}$. Then, z_1 publishes $\{l_1{}^{z_1}, r_1{}^{z_1}\}$ in DB. The rest of the nodes agree that $r_1{}^{z_1}$ has the minimum distance to $r_1{}^{z_0}$; that is, $r_1{}^{z_0} \sim r_1{}^{z_1}$, and z_1 wins the first round.

The notation $k_j{}^{z_i}$ will be used in the following discussion to indicate that the key (k_j) belongs to node z_i. To simplify the explanations, we will assume that the winning nodes appear in order, e.g., $z_0, z_1 \ldots$, and we only discuss the first two rounds:

- Round 0 (registration phase): Using his numbers, $\{{k_0}^{z_0}, {k_1}^{z_0}, {k_2}^{z_0}, {x_0}^{z_0}, {y_0}^{z_0}\}$, the root player, denoted as z_0, registers $\{{c_0}^{z_0}, {l_0}^{z_0}, {r_0}^{z_0}\}$ in the public DB, where ${c_0}^{z_0} = {k_0}^{z_0} \oplus {k_1}^{z_0}$, ${l_0}^{z_0} = {x_0}^{z_0} \oplus {k_0}^{z_0} \oplus {k_2}^{z_0}$ and ${r_0}^{z_0} = {y_0}^{z_0} \oplus {k_1}^{z_0} \oplus {k_2}^{z_0}$. In addition, z_0 stores ${h_0}^{z_0}$ in DB, where ${h_0}^{z_0} = {x_0}^{z_0} \oplus {y_0}^{z_0}$. But z_0 computes ${h_0}^{z_0}$, so that is the root of his Merkle tree. Then, all players select their own set of numbers, for example, z_1 chooses $\{{k_0}^{z_1}$, ${k_1}^{z_1}$, ${k_2}^{z_1}$, ${x_0}^{z_1}$, and ${y_0}^{z_1}\}$, so that they match the public keys of z_0.
- Round 1: All nodes, for example, z_1 using $\{{x_0}^{z_1}, {y_0}^{z_1}, {x_1}^{z_1}, {y_1}^{z_1}\}$, compute ${h_0}^{z_1} = {x_0}^{z_1} \oplus {y_0}^{z_1}$ and ${h_1}^{z_1} = {x_1}^{z_1} \oplus {y_1}^{z_1}$. But z_1 chooses his numbers, so that ${h_1}^{z_1}$ yields the root of his Merkle tree. Then, z_1 publishes $\{{h_0}^{z_1}, {h_1}^{z_1}\}$ in DB to the rest of the nodes. Now, node z_0 publishes $\{{l_1}^{z_0}, {r_1}^{z_0}\}$ in DB, where ${l_1}^{z_0} = {k_1}^{z_0} \oplus {k_3}^{z_0} \oplus {x_0}^{z_0} \oplus {x_1}^{z_0}$ and ${r_1}^{z_0} = {y_1}^{z_0} \oplus {k_2}^{z_0} \oplus {k_3}^{z_0}$. All nodes, for example, z_1 using ${k_3}^{z_1}$, obtain ${l_1}^{z_1} = {k_1}^{z_1} \oplus {k_3}^{z_1} \oplus {x_0}^{z_1} \oplus {x_1}^{z_1}$ and ${r_1}^{z_1} = {y_1}^{z_1} \oplus {k_2}^{z_1} \oplus {k_3}^{z_1}$. But z_1 chooses ${k_3}^{z_1}$, so that ${l_1}^{z_1} = {l_1}^{z_0}$ (as illustrated in Figure 9). Then, z_1 publishes $\{{l_1}^{z_1}, {r_1}^{z_1}\}$ in DB to the rest of the nodes. If the nodes (or the majority of them) agree that ${r_1}^{z_1}$ has the minimum distance to ${r_1}^{z_0}$, then z_1 wins the first round. Now, DB removes the auxiliary data from the other nodes but permanently stores $\{{l_1}^{z_1}, {r_1}^{z_1}\}$, and $\{{h_0}^{z_1}, {h_1}^{z_1}\}$ to enable all nodes to verify z_1 because ${l_1}^{z_1} \oplus {r_1}^{z_1} = {h_0}^{z_1} \oplus {h_1}^{z_1}$. Before the next round takes place, each node, say z_2, selects its numbers again, ensuring that ${h_1}^{z_2} = {h_1}^{z_1}$, where ${h_1}^{z_1}$ is the root of the Merkle tree that belongs to z_1. However, ${h_2}^{z_2}$ is the root of the Merkle tree of the z_2 node.

The public XOR chain contains three numbers: $\{l_j, r_j, h_j\}$, where the term h_j is the root of the Merkle tree and the chaining process is $h_{j-1} = h_j$, as indicated in Table 5. Assuming a 256-bit hash function, it yields 768 bits for each entry.

Table 5. The XOR chain is conformed by the succession $\{{l_0}^{z_0}, {r_0}^{z_0}, {h_0}^{z_0}\}, \{{l_1}^{z_1}, {r_1}^{z_1}, {h_1}^{z_1}\}, \ldots \{{l_j}^{z_j}, {r_j}^{z_j}, {h_j}^{z_j}\}$, where ${h_0}^{z_0} = {h_0}^{z_1}$, ${h_1}^{z_1} = {h_1}^{z_2} \ldots$.

j	l_j	r_j	h_{j-1}		h_j
0	${l_0}^{z_0}$	${r_0}^{z_0}$	—		${h_0}^{z_0}$
				$\swarrow$	
1	${l_1}^{z_1}$	${r_1}^{z_1}$	${h_0}^{z_1}$		${h_1}^{z_1}$
				$\swarrow$	
2	${l_2}^{z_2}$	${r_2}^{z_2}$	${h_1}^{z_2}$		${h_2}^{z_2}$
				$\swarrow$	
$\vdots$	$\vdots$	$\vdots$			$\vdots$
j	${l_j}^{z_j}$	${r_j}^{z_j}$	${h_{j-1}}^{z_j}$		${h_j}^{z_j}$
				$\swarrow$	

4. Data Encryption

The secret exchange of sensitive information between remote users is achieved using data encryption techniques. Our encryption method resembles the cipher block chaining (CBC) mode, which is frequently employed by symmetric block ciphers. A mode of operation is a method for concealing patterns in plaintext by encrypting datasets larger than a block size. This measure prevents statistical frequency attacks, which could be found in the original plaintext.

In Section 4.1, we will describe the cipher block chaining (CBC) mode. This chaining mode will allow us to establish a baseline for our XOR encryption scheme. The 3-encryption algorithm is introduced in Section 4.2.

4.1. CBC Mode Encryption

Figure 10 illustrates the basic operation of the cipher block chaining (CBC) mode mechanism: in the first round, an initialization vector (IV) and the data block are XORed, and then encrypted with the chosen encryption algorithm [38,58]. This encrypted output replaces the IV in the next round. Therefore, if a plaintext block appears again, CBC ensures that a different ciphertext is produced because the encryption depends on the previous plaintext blocks.

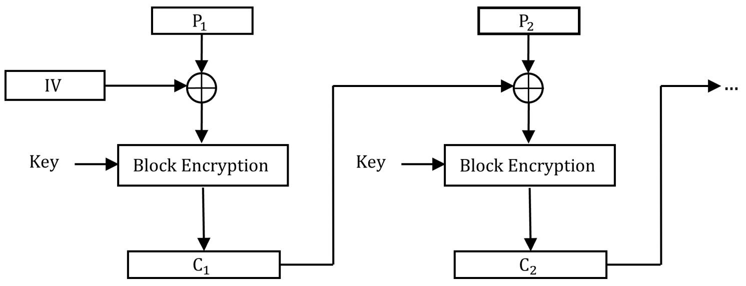


Figure 10. CBC mode: Before the encryption process, the plaintext block and the previous ciphertext block are passed to the XOR function. As a result, each round generates a cipher block that is dependent on the previous plaintext blocks.

4.2. 3-Encryption

Alice wants to send secret messages to Bob via the triple XOR encryption cancellation rule depicted in Figure 2. To achieve perfect secrecy, we will use this model but replace the XOR operation with integer addition. We shall describe the algorithm first, followed by a discussion of its security.

Let us assume Alice and Bob share an initial secret key because they have possibly performed a secret key establishment algorithm. Actually, they share three positive integers k_0, k_1, k_2, then users proceed to compute $\{k_1 - k_0, k_2 - k_0, k_2 - k_1\}$. In this scenario, there is no database of any user, so we eliminated the term d_j from Equation (7). As a result, we introduce Equation (10)

$$\begin{aligned} l_j &= m_j + k_{j+2} - k_j \\ r_j &= m_{j-1} + k_{j+2} - k_{j+1} \\ c_j &= r_{j-1} \end{aligned} \tag{10}$$

where messages and keys are positive integers. Suppose Alice starts sending the first encrypted message m_0 to Bob. Users run the protocol depicted in Figure 11. The rounds are detailed in Table 6, and Figure 12 depicts the message exchange between Alice and Bob.

In the round denoted with sub-index j, Bob applies the deciphering rule specified by Equation (11)

$$m_j = l_j - r_j - r_{j-1} + m_{j-1} + m_{j-2} \tag{11}$$

where the initial round ($j = 0$) is run as depicted in Figure 11.

4.3. Perfect Secrecy

Our goals to achieve perfect secrecy are enlisted below.

1. Every encryption key must be distinct from any previous key.
2. The number of available keys is greater than or equal to the number of messages in the system.
3. Every key is chosen randomly and the probability must be the same for all keys.

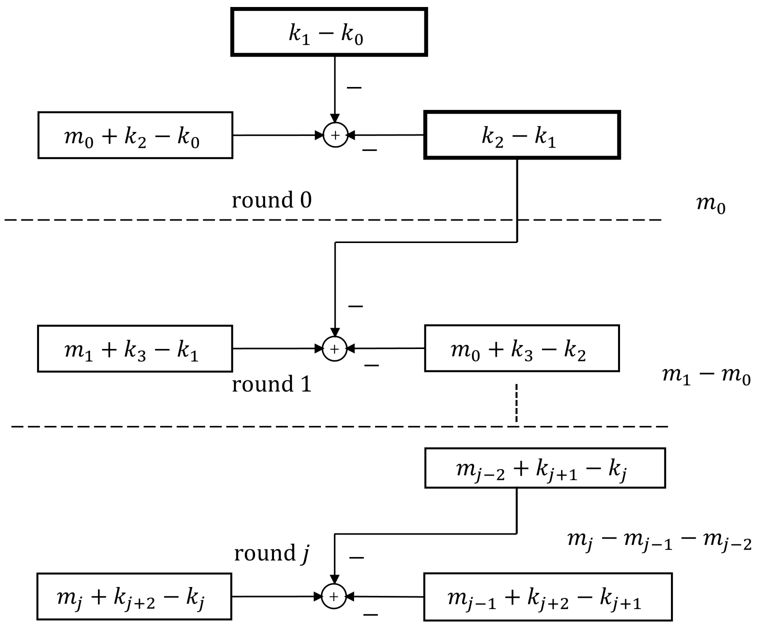
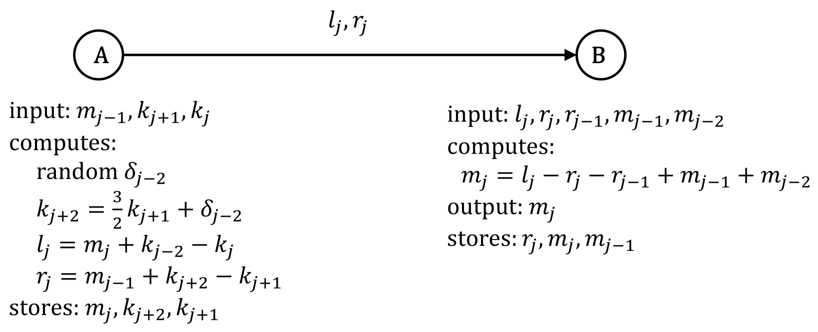


Figure 11. Boxes with bold outline imply that these numbers are private and will not be transmitted over the public channel.

Table 6. Protocol execution for rounds $0 \dots j$. The terms m_j, c_j, l_j, and r_j are shown after each execution round. The last column shows the result of $l_j - r_j - r_{j-1}$.

Round	$l_j = m_j + k_{j+2} - k_j$	$r_j = m_{j-1} + k_{j+2} - k_{j+1}$	$c_j = m_{j-2} + k_{j+1} - k_j$	$l_j - r_j - r_{j-1}$
0	$m_0 + k_2 - k_0$	$k_2 - k_1$	$k_1 - k_0$	m_0
1	$m_1 + k_3 - k_1$	$m_0 + k_3 - k_2$	$k_2 - k_1$	$m_1 - m_0$
2	$m_2 + k_4 - k_2$	$m_1 + k_4 - k_3$	$m_0 + k_3 - k_2$	$m_2 - m_1 - m_0$
3	$m_3 + k_5 - k_3$	$m_2 + k_5 - k_4$	$m_1 + k_4 - k_3$	$m_3 - m_2 - m_1$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
j	$m_j + k_{j+2} - k_j$	$m_{j-1} + k_{j+2} - k_{j+1}$	$m_{j-2} + k_{j+1} - k_j$	$m_j - m_{j-1} - m_{j-2}$



l_j, r_j
A
B
input: m_{j-1}, k_{j+1}, k_j
computes:
random δ_{j-2}
$k_{j+2} = \frac{3}{2} k_{j+1} + \delta_{j-2}$
$l_j = m_j + k_{j-2} - k_j$
$r_j = m_{j-1} + k_{j+2} - k_{j+1}$
stores: m_j, k_{j+2}, k_{j+1}
input: $l_j, r_j, r_{j-1}, m_{j-1}, m_{j-2}$
computes:
$m_j = l_j - r_j - r_{j-1} + m_{j-1} + m_{j-2}$
output: m_j
stores: r_j, m_j, m_{j-1}

Figure 12. Message exchange in round j of the XOR encrypting algorithm.

Let us proceed with the first point. In Table 6, we can rewrite the $j-$term of column 2 as $l_j = m_j + k_j{}'$, where $k_j{}'$ is the encrypting key computed as $k_j{}' = k_{j+2} - k_j$. Similarly, for column 3, we can establish that $r_j = m_{j-1} + k_j{}''$, where $k_j{}''$ is derived as $k_j{}'' = k_{j+2} - k_{j+1}$. Now, each encrypting key ($k_j{}'$, $k_j{}''$) must be distinct from every previous key. We will accomplish this task by employing an inequality relationship, in which the value of each key is greater than the preceding one. Thus, in column 2 of Table 6, we must guarantee that $k_2 - k_0 < k_3 - k_1 < k_4 - k_2 < \ldots < k_{k_{j+1}} - k_{j-1} < k_{k_{j+2}} - k_j$. It implies that $k_j > k_{j-1} + k_{j-2} - k_{-3}$. Also, in column 3 of Table 6, we must verify that $k_2 - k_1 < k_3 - k_2 < k_4 - k_3 < \ldots < k_{k_{j+1}} - k_j < k_{k_{j+2}} - k_{j+1}$, which leads us to $k_j > 2k_{j-1} - k_{j-1}$. Then, by combining both results, we obtain Equation (12)

$$k_j > \frac{3}{2}k_{j-1} - \frac{1}{2}k_{j-3} \tag{12}$$

which is equally valid if $k_j > \frac{3}{2}k_{j-1}$. Thus, Equation (13) allows us to achieve the first enlisted point

$$k_j = \frac{3}{2}k_{j-1} + \delta_j \tag{13}$$

where δ_j is a positive random number computed in each round.

The second point is strongly related to the first because using a distinct key for each message guarantees that the key space and message space are comparable. Now, let us address the third point. We will assume that the number of keys in the system is j and the key space is product jN, where N is a sufficiently large number. In each round, the key is chosen from a window containing N keys. Then, the probability of choosing k_j is $P(K = k_j) = \frac{1}{N}$ for $j \geq 0$. What follows is the proof of perfect secrecy, which can be found in Appendix A of this document.

However, because of the structure of our cryptosystem, if a message happens to be repeated in the next round, the previous one can be deduced, so in the last column of Table 6, $m_{j-1} - m_{j-1} - m_{j-2} = m_{j-2}$. This is due to the non-existence of a permutation or data dispersion process in our system, which, on the other hand, can be done in image encryption and blockchain encryption approaches.

At this point, we realized that interleaving a random integer between two consecutive cleartext messages is a convenient way for ensuring secrecy. So, in round 0, instead of sending m_0, the random x_0 should be sent. In round 1, Alice inserts the random x_1 and then she sends m_1. Moreover, Alice could transmit only one term rather than two as depicted in Figure 11 over the public channel. So, Alice computes the term $l_j - r_j - r_{j-1}$ and then she sends it to Bob.

Thus, in round j we have that $x_j - x_{j-1} - m_{j-1}$ precedes the term $m_j - m_{j-1} - x_j$ and the last column in Table 6 becomes $m_j - m_{j-1} - x_j$. Furthermore, as previously established, the following inequalities must be met in order to maintain perfect secrecy $x_{j-1} - x_{j-2} < x_j - x_{j-1}$ which reduces to $x_j > 2x_{j-1} - x_{j-2}$ or simply $x_j > 2x_{j-1}$. Consider the following deciphering rule comparison, where the term in parenthesis denotes a single term (stored or received from the channel):

— $m_j = l_r - r_j - r_{j-1} + m_{j-1} + x_j$ where $x_j = (x_j - x_{j-1} - m_{j-1}) + x_{j-1} + m_{j-1}$ requires 6 additions and 4 terms to be stored: $(x_j - x_{j-1} - m_{j-1}), x_{j-1}, m_{j-1}, r_{j-1}$.
— $m_j = (m_j - m_{j-1} - x_j) + m_{j-1} + x_j$ where $x_j = (x_j - x_{j-1} - m_{j-1}) + x_{j-1} + m_{j-1}$ requires 4 additions and 3 terms to be stored: $(x_j - x_{j-1} - m_{j-1}), x_{j-1}, m_{j-1}$.

5. Discussion and Future Work

Although our system has not yet been experimentally implemented, we will highlight some features of our digital signature, block chaining, and encryption schemes, in comparison to some of the current pre-quantum and post-quantum cryptosystems. Among all the features of our system, the most notable is its simplicity, as it only requires XOR operations, hash, and integer addition. However, in this section, we will identify and discuss the drawbacks and challenges that we might face during its implementation.

1. Digital signature: To avoid analysis by Grover's quantum search technique, we shall assume that the hash code must be at least 256 bits long. Thus, the size of the private key, $\{k_0, k_1, k_2\}$, achieves $3 \times 256 = 768$ bits. In addition, the size of the public key, $\{x_0 \oplus k_0 \oplus k_2\}, \{y_0 \oplus k_1 \oplus k_2\}, \{k_0 \oplus k_1\}$ reaches $3 \cdot 256 = 768$ bits. These key sizes are really small compared to current public key cryptosystems. Figure 5 in

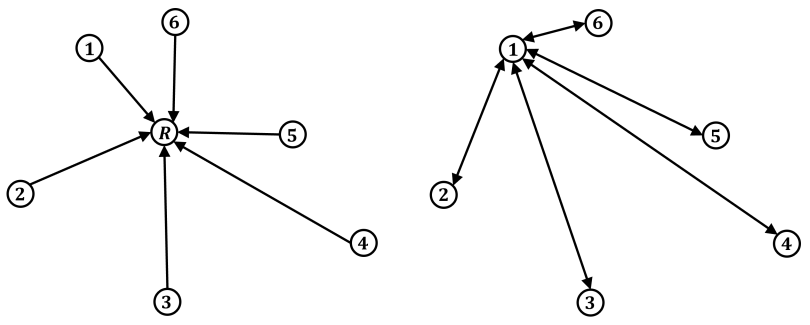
Section 2.3 shows the storage requirements: 1024 bits for Alice, Bob occupies 512 bits, and DB requires 256 bits. The signature process uses three random numbers, five XOR operations, and one hash computation. Finally, signature verification needs four XOR operations, one hash, and one comparison. The main drawback of our method involves the need for a central node, which can introduce delays in the signature process.

2. Blockchain: In the blockchain system [40], miners calculate and maintain a unified chain of all transactions on the network. In the distributed database, a table entry is 768 bits in size. In contrast, the table entry size is 8192 bits in the HMAC chain and hash chain algorithms. In this scenario, every user calculates and maintains his own independent chain of transactions. The table entry size in the XOR chain introduced here is 768 bits. Similar to blockchain, network nodes compute and store copies of a unified chain of transactions. However, some of the major advantages of our scheme over blockchain are the following: (1) it does not require proof of work (PoW), and (2) it is immune to quantum cryptanalysis. The comparison of such parameters is given in Table 7.
3. The 3-encryption: Figure 12 in Section 4.2 shows the storage prerequisites for 3-encryption. If we assume that both the message segment and key size are 256 bits, Alice requires 768 bits and Bob needs the same. The encryption process demands the generation of one random number, five (adding) operations, and one multiplication. Decryption takes four (adding) operations. The encryption method holds promise for exceptional execution performance since it does not necessitate numerous rounds of substitution and permutation, unlike the symmetric CBC chained mode. However, as previously stated, our system lacks a data permutation or dispersion mechanism; thus, if a message is repeated in the next round, the previous one can be inferred.

Table 7. Block chaining schemes are compared below. The sizes are written in bits. PQ stands for post-quantum while PoW stands for proof of work.

	Blockchain	**Hash Chain**	**HMAC Chain**	**This Work**
Table Size Entry	768	8192	8192	768
PQ	no	yes	yes	yes
PoW	yes	no	no	no

Figure 13 depicts a possible scenario for the implementation of our system: a sensor network requires message authentication and data encryption. Case (a) shows the digital signature mode, which requires a central node (R) for the execution of the XOR signature. In (b), the nodes have a secret initial key that they use to encrypt the data using the XOR encryption algorithm.



(**a**) Digital Signature mode (**b**) Encryption mode

Figure 13. In scenario (**a**), with the help of central node R, any pair of nodes can exchange signed data. In (**b**), Node 1 has an initial secret key with the other nodes in the network, so it can maintain encrypted communications with them.

For the initial encryption tests, it is feasible to use a basic pre-shared key scheme. However, it will be necessary to choose a secure method for the shared secret key exchange. On the other hand, it is a challenging task to maintain a secret key with each node in the network. At this point, it would be necessary to analyze whether the central node could facilitate this task. The digital signature scheme requires the incorporation of a cryptographic hash function, which will be chosen by prioritizing security and efficiency.

The signature mode depicted in Figure 13 can be used for block chaining. Although storage is a typical requirement of blockchain, it is still a concern of the system since a set of bits is accumulated in each round of the signature process. Likewise, the scheme requires an active node during the signature verification. If the node fails, system transactions could not be carried out. To reverse this situation, we visualize the following alternatives:

1. Keep a distributed copy of the database among the users of the system, which is the approach used by the blockchain system.
2. Offline operation. The protocol operates without requiring intervention from the central node, which would guarantee the continuity of the system in the event of failures.

We reserve for future investigation the inquiry of whether the 3-encryption system can operate in stream encryption mode for video/audio traffic applications and will analyze its efficiency with respect to the current CFB and OFB methods in streaming mode.

6. Conclusions

The possibility of establishing Shannon's perfect secrecy in a technologically workable way has not been investigated in sufficient depth. In this paper, we show that our method provides perfect secrecy. For this purpose, the triple cancellation rule was introduced here, which is easy to verify and evaluate.

Digital signature and 3-encryption are the two modes of the fundamental algorithm that have been covered. Digital signatures depend on a central node to verify signatures. Message encryption does not require a central node, but it uses a shared secret key, which can be pre-loaded on nodes or exchanged using a key exchange algorithm.

We have found that the digital signature mode can be used to define a block chaining mode that does not require proof of work (PoW). This method is based on Crypto Bingo, a similar algorithm that we introduced in this work. The proposed scheme, like blockchain, requires 768 bits of storage per transaction. However, it is well-known that blockchain and the algorithms currently used for digital signatures do not support quantum cryptanalysis. As a result, given the challenges that the development of quantum computers imposes on cryptographic security, we believe that our scheme represents a secure, efficient, and simple alternative to performing digital signatures, data encryption, and blockchain.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

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

Appendix A. Perfect Secrecy Proof

We base our proof on the general demonstration of perfect secrecy given in [59]. We assume plaintexts and keys with fixed (non-variable) length. We must prove that

$$P\big(M = m_j | E = e_j\big) = P\big(M = m_j\big)$$

for each pair of m_j, e_j. Every key is generated according to Equation (13), where δ_{j+1} is a random number. As a result, keys are independent each other and we can add their probabilities, therefore

$$P\big(E = e_j\big) = \sum_k P\big(E = e_j \cap K = k_j\big)$$

The message and the key are chosen independently, then

$$P(E = e_j \cap K = k_j) = P(M = e_j - k_j \cap K = k_j)$$

$$= P(M = e_j - k_j)P(K = k_j)$$

$$= P(M = e_j - k_j)\frac{1}{N}$$

The probability of a key is $P(K = k) = \frac{1}{N}$, then $M = e_j - k_j$ goes through all possible messages, so

$$\sum_k P(M = e_j - k_j) = 1$$

then we obtain

$$P(E = e_j) = \sum_k P(E = e_j \cap K = k_j)$$

$$= \frac{1}{N}\sum_k P(M = e_j - k_j) = \frac{1}{N}$$

The conditional probability definition establishes that

$$P(M = m_j | E = e_j)P(E = e_j) = P(E = e_j \cap M = m_j)$$

and the independence of K and M produce

$$= P(K = e_j - m_j \cap M = m_j) = P(K = e_j - m_j)P(M = m_j)$$

multiplying by N because $P(E = e_j) = \frac{1}{N} = P(K = e_j - m_j)$ leads to

$$= P(M = m_j | E = e_j) = P(M = m_j)$$

which implies that the XOR-encryption method achieves perfect secrecy.

References

1. Nielsen, M.A.; Chuang, I.L. *Quantum Computation and Quantum Information*; Cambridge University Press: Cambridge, UK, 2010.
2. Dattani, N.S.; Bryans, N. Quantum factorization of 56153 with only 4 qubits. *arXiv* **2014**, arXiv:1411.6758.
3. Dridi, R.; Alghassi, H. Prime factorization using quantum annealing and computational algebraic geometry. *arXiv* **2016**, arXiv:1604.05796.
4. Shor, P.W. Algorithms for quantum computation: Discrete logarithms and factoring. In Proceedings of the 35th Annual Symposium on Foundations of Computer Science, Santa Fe, NM, USA, 20–22 November 1994; pp. 124–134.
5. 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; pp. 212–219.
6. Nagaraj, N.; Vaidya, V.; Vaidya, P.G. Re-visiting the One-Time Pad. *arXiv* **2005**, arXiv:cs/0508079.
7. Damgård, I.; Pedersen, T.B.; Salvail, L. A quantum cipher with near optimal key-recycling. In *Proceedings of the Advances in Cryptology–CRYPTO 2005: 25th Annual International Cryptology Conference, Santa Barbara, CA, USA, 14–18 August 2005. Proceedings 25*; Springer: Cham, Switzerland, 2005; pp. 494–510.
8. Baptista, M. Cryptography with chaos. *Phys. Lett. A* **1998**, *240*, 50–54. [CrossRef]
9. Jakimoski, G.; Kocarev, L. Chaos and cryptography: Block encryption ciphers based on chaotic maps. *IEEE Trans. Circuits Syst. I Fundam. Theory Appl.* **2001**, *48*, 163–169. [CrossRef]
10. Dachselt, F.; Schwarz, W. Chaos and cryptography. *IEEE Trans. Circuits Syst. I Fundam. Theory Appl.* **2001**, *48*, 1498–1509. [CrossRef]
11. Nagaraj, N. One-Time Pad as a nonlinear dynamical system. *Commun. Nonlinear Sci. Numer. Simul.* **2012**, *17*, 4029–4036. [CrossRef]
12. Ajtai, M. Generating hard instances of lattice problems. In Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, Philadelphia, PA, USA, 22–24 May 1996; pp. 99–108.

13. Hoffstein, J.; Pipher, J.; Silverman, J.H. NTRU: A ring-based public key cryptosystem. In *Proceedings of the International Algorithmic NUMBER Theory Symposium*; Springer: Berlin/Heidelberg, Germany, 1998; pp. 267–288.
14. Buchmann, J.; Dahmen, E.; Hülsing, A. XMSS-a practical forward secure signature scheme based on minimal security assumptions. In *Proceedings of the Post-Quantum Cryptography: 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, 29 November–2 December 2011. Proceedings 4*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 117–129.
15. Bernstein, D.J.; Hopwood, D.; Hülsing, A.; Lange, T.; Niederhagen, R.; Papachristodoulou, L.; Schneider, M.; Schwabe, P.; Wilcox-O'Hearn, Z. SPHINCS: Practical stateless hash-based signatures. In *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 368–397.
16. McEliece, R.J. A public-key cryptosystem based on algebraic. *Coding Thv* **1978**, *4244*, 114–116.
17. Niederreiter, H. Knapsack-type cryptosystems and algebraic coding theory. *Prob. Contr. Inform. Theory* **1986**, *15*, 157–166.
18. Matsumoto, T.; Imai, H. Public quadratic polynomial-tuples for efficient signature-verification and message-encryption. In *Proceedings of the Advances in Cryptology—EUROCRYPT'88: Workshop on the Theory and Application of Cryptographic Techniques Davos, Switzerland, 25–27 May 1988 Proceedings 7*; Springer: Berlin/Heidelberg, Germany, 1988; pp. 419–453.
19. Ding, J.; Schmidt, D. Rainbow, a new multivariable polynomial signature scheme. In *Proceedings of the International Conference on Applied Cryptography and Network Security*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 164–175.
20. Jao, D.; De Feo, L. Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies. In *Proceedings of the Post-Quantum Cryptography: 4th International Workshop, PQCrypto 2011, Taipei, Taiwan, 29 November–2 December 2011. Proceedings 4*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 19–34.
21. Standard, A.E. Federal Information Processing Standards Publication 197. FIPS PUB. 2001; pp. 3–46. Available online: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf (accessed on 12 October 2023).
22. Campagna, M.; Hardjono, T.; Pintsov, L.; Romansky, B.; Yu, T. Kerberos revisited quantum-safe authentication. In Proceedings of the ETSI Quantum-Safe-Crypto Workshop, Nice, France, 26–27 September 2013; pp. 26–27.
23. Bernstein, D.J.; Lange, T. Post-quantum cryptography. *Nature* **2017**, *549*, 188–194. [CrossRef]
24. Alagic, G.; Apon, D.; Cooper, D.; Dang, Q.; Dang, T.; Kelsey, J.; Lichtinger, J.; Miller, C.; Moody, D.; Peralta, R.; et al. *Status Report on the Third Round of the Nist Post-Quantum Cryptography Standardization Process*; US Department of Commerce, NIST: Gaithersburg, MD, USA, 2022.
25. Laboratory, I.T. PQC Standardization Process: Third Round Candidate Announcement. 2020. Available online: https://csrc.nist.gov/news/2020/pqc-third-round-candidate-announcement (accessed on 12 October 2023).
26. Chen, L.; Chen, L.; Jordan, S.; Liu, Y.K.; Moody, D.; Peralta, R.; Perlner, R.; Smith-Tone, D. *Report on Post-Quantum Cryptography*; US Department of Commerce, National Institute of Standards and Technology: Gaithersburg, MD, USA, 2016; Volume 12.
27. Persichetti, E. NIST Round 3 Finalists. 2020. Available online: https://pqc-wiki.fau.edu/w/Special:DatabaseHome (accessed on 12 October 2023).
28. Castryck, W.; Decru, T. An Efficient Key Recovery Attack on SIDH (Preliminary Version). Cryptology ePrint Archive 2022. Available online: https://eprint.iacr.org/2022/975 (accessed on 12 October 2023).
29. Beullens, W. Breaking Rainbow Takes a Weekend on a Laptop. Cryptology ePrint Archive, Paper 2022/214. 2022. Available online: https://eprint.iacr.org/2022/214 (accessed on 12 October 2023).
30. Beullens, W. Improved cryptanalysis of UOV and rainbow. In *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*; Springer International Publishing: Cham, Switzerland, 2021; pp. 348–373.
31. Rivest, R.L.; Shamir, A.; Adleman, L. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM* **1978**, *21*, 120–126. [CrossRef]
32. PUB, F. Digital Signature Standard (DSS). FIPS PUB. 2000; pp. 186–192. Available online: https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf (accessed on 12 October 2023).
33. Moody, D.; Alagic, G.; Apon, D.C.; Cooper, D.A.; Dang, Q.H.; Kelsey, J.M.; Liu, Y.K.; Miller, C.A.; Peralta, R.C.; Perlner, R.A.; et al. *Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process*; US Department of Commerce, NIST: Gaithersburg, MD, USA, 2020.
34. Merkle, R.C. *Secrecy, Authentication, and Public Key Systems*; Stanford University: Stanford, CA, USA, 1979.
35. 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. [CrossRef]
36. Lizama-Perez, L.A. Digital signatures over hash-entangled chains. *SN Appl. Sci.* **2019**, *1*, 1568. [CrossRef]
37. Schneier, B. Description of a new variable-length key, 64-bit block cipher (Blowfish). In *Proceedings of the International Workshop on Fast Software Encryption*; Springer: Berlin/Heidelberg, Germany, 1993; pp. 191–204.
38. Rogaway, P. Evaluation of Some Blockcipher Modes of Operation. Cryptography Research and Evaluation Committees (CRYPTREC) for the Government of Japan. 2011. Available online: https://www.cs.ucdavis.edu/~rogaway/papers/modes-cryptrec.pdf (accessed on 12 October 2023).
39. Bujari, D.; Aribas, E. Comparative analysis of block cipher modes of operation. In Proceedings of the International Advanced Researches & Engineering Congress, Osmaniye, Turkey, 16–18 November 2017; pp. 1–4.
40. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. Decentralized Business Review. 2008, p. 21260. Available online: https://assets.pubpub.org/d8wct41f/31611263538139.pdf (accessed on 12 October 2023).

41. Johar, S.; Ahmad, N.; Asher, W.; Cruickshank, H.; Durrani, A. Research and applied perspective to blockchain technology: A comprehensive survey. *Appl. Sci.* **2021**, *11*, 6252. [CrossRef]
42. Kearney, J.J.; Perez-Delgado, C.A. Vulnerability of blockchain technologies to quantum attacks. *Array* **2021**, *10*, 100065. [CrossRef]
43. Vujičić, D.; Jagodić, D.; Ranđić, S. Blockchain technology, bitcoin, and Ethereum: A brief overview. In Proceedings of the 2018 17th International Symposium Infoteh-Jahorina (Infoteh), East Sarajevo, Bosnia and Herzegovina, 21–23 March 2018; pp. 1–6.
44. Paulavičius, R.; Grigaitis, S.; Igumenov, A.; Filatovas, E. A decade of blockchain: Review of the current status, challenges, and future directions. *Informatica* **2019**, *30*, 729–748. [CrossRef]
45. Papageorgiou, O.; Sedlmeir, J.; Fridgen, G.; Vlachos, I.; Kostopoulos, N.; Damvakeraki, T.; Noszek, Z.; Papoutsoglou, I.; Anania, A.; Belotti, M.; et al. *Energy Efficiency of Blockchain Technologies*; European Union Blockchain Observatory & Forum. 2021. Available online: https://www.eublockchainforum.eu/sites/default/files/reports/Energy%20Efficiency%20of%20Blockchain%20Technologies_1.pdf (accessed on 12 October 2023).
46. Fernandez-Carames, T.M.; Fraga-Lamas, P. Towards post-quantum blockchain: A review on blockchain cryptography resistant to quantum computing attacks. *IEEE Access* **2020**, *8*, 21091–21116. [CrossRef]
47. Edwards, M.; Mashatan, A.; Ghose, S. A review of quantum and hybrid quantum/classical blockchain protocols. *Quantum Inf. Process.* **2020**, *19*, 184. [CrossRef]
48. Ruggeri, C. Quantum Key Distribution in Softwarised Networks. Ph.D. Thesis, Politecnico di Torino, Turin, Italy, 2020.
49. Ikeda, K. qBitcoin: A peer-to-peer quantum cash system. In *Proceedings of the Intelligent Computing: Proceedings of the 2018 Computing Conference, Volume 1*; Springer International Publishing: Cham, Switzerland, 2019; pp. 763–771.
50. Gottesman, D.; Chuang, I. Quantum digital signatures. *arXiv* **2001**, arXiv:quant-ph/0105032.
51. Shannon, C.E. Communication theory of secrecy systems. *Bell Syst. Tech. J.* **1949**, *28*, 656–715. [CrossRef]
52. Feutrill, A.; Roughan, M. A Review of Shannon and Differential Entropy Rate Estimation. *Entropy* **2021**, *23*, 1046. [CrossRef] [PubMed]
53. Martin, K.M. Everyday cryptography. *The Australian Mathematical Society*; Oxford University Press: Oxford, UK, 2012; pp. 231–234.
54. Shimeall, T.; Spring, J. *Introduction to Information Security: A Strategic-Based Approach*; Newnes: Oxford, UK, 2013.
55. Lizama-Pérez, L.A. Digital signatures over HMAC entangled chains. *Eng. Sci. Technol. Int. J.* **2021**, *32*, 101076. [CrossRef]
56. Krawczyk, H.; Canetti, R.; Bellare, M. HMAC: Keyed-Hashing for Message Authentication. 1997. Available online: https://www.rfc-editor.org/rfc/rfc2104 (accessed on 12 October 2023).
57. Yan, B.; Tan, Z.; Wei, S.; Jiang, H.; Wang, W.; Wang, H.; Luo, L.; Duan, Q.; Liu, Y.; Shi, W.; et al. Factoring integers with sublinear resources on a superconducting quantum processor. *arXiv* **2022**, arXiv:2212.12372.
58. Ehrsam, W.F.; Meyer, C.H.; Smith, J.L.; Tuchman, W.L. Message Verification and Transmission Error Detection by Block Chaining. U.S. Patent 4,074,066, 14 February 1978.
59. Trappe, W. *Introduction to Cryptography with Coding Theory*; Pearson Education: London, UK, 2020.