electronics

MDPI

Article

A Memorable Communication Method Based on Cryptographic Accumulator

Wenbao Jiang *, Yongpan Wang and Shuai Ye

Department of Information Security, Beijing Infomation Science and Technology University, Beijing 100192, China; 2021020903@bistu.edu.cn (Y.W.); 2020020737@bistu.edu.cn (S.Y.)
* Correspondence: jiangwenbao@tsinghua.org.cn

Abstract: The traditional Internet has many security problems. It is difficult to guarantee the authenticity, integrity, and synchronization of message transmission, and it lacks a message-traceability mechanism, which is caused by its performance-oriented design. To address these problems, this paper proposes a memorable communication method based on cryptographic accumulators. In this method, both parties in the communication can verify the message data sent and received arbitrarily by virtue of the memory value. As long as a simple memory value comparison is performed, the strong consistency of all message data can be ensured. This method has the security advantages of synchronization, verification, traceability, and non-tamperability, as well as the performance advantages brought by batch signature and verification. In this paper, the memorable communication model, the memory function, and the memorable communication process are designed, and theoretical analysis shows that the memorable communication method has synchronization and traceability and can realize batch signature and authentication. In addition, a chain-key can be constructed based on a memory value to achieve key per-packet updating. Comparative analysis shows the transmission efficiency, traceability efficiency, and security performance of the memorable communication method.

Keywords: information security; memorable communication method; cryptographic accumulator; endogenous security; batch signature; chain-key

Citation: Jiang, W.; Wang, Y.; Ye, S. A Memorable Communication Method Based on Cryptographic Accumulator. *Electronics* **2024**, *13*, 1081. https://doi.org/10.3390/electronics13061081

Academic Editors: Zbigniew Kotulski and Dimitra I. Kaklamani

Received: 25 December 2023
Revised: 18 February 2024
Accepted: 13 March 2024
Published: 14 March 2024

1. Introduction

The Internet, built on the TCP/IP protocol, is pivotal in modern society, driving the era of the Internet of Everything with technologies like big data, cloud computing, and blockchain. However, security challenges persist, including source address spoofing and DDoS attacks. Traditional solutions struggle, necessitating innovations with inherent security features [1].

Initially, communication networks were primarily designed for performance, often overlooking data integrity and authenticity verification. Consequently, transmitted content becomes vulnerable to forgery and tampering by malicious actors, making it challenging to trace attackers. To address these concerns, various scholars have proposed solutions such as the Host Identity Protocol (HIP) [2], which separates the host identifier from the IP address. While HIP enhances identity identification, it falls short of ensuring message transmission security. Similarly, approaches like Accountable and Private Internet Protocol [3] and Accountable Internet Protocol [4] attempt to blend message auditability and privacy but lack robust security features, allowing senders to deny message ownership.

Existing technologies like IPsec offer partial solutions but fail to address issues like non-repudiation and message synchronization [5], particularly challenging in resource-constrained IoT environments. This highlights the need for novel approaches that not only ensure security but also address practical application demands.

Cryptographic accumulators emerge as a promising solution [6], offering concise commitment values for collections of elements. Their versatility extends to various applications, including anonymous authentication schemes, group signatures, and stateless

blockchains. However, current implementations still lack clarity in their practical application and fail to fully address specific application demands. Cryptographic accumulator can be divided into three categories according to the different security assumptions used: RSA accumulator [6–10], bilinear pair accumulator [11–14] and hash function accumulator [15]. Additionally, innovations like the message hash chain [16,17], akin to cryptographic accumulators, further contribute to message security. Proposed by Han Mingxuan et al., message hash chains iteratively hash transmitted message values, ensuring secure transmission [18,19].

In this paper, our contributions are as follows:

This paper proposes a memorable communication method based on cryptographic accumulators. It designs a memorable communication model, memory function, and communication process. The communication parties can generate corresponding synchronization parameters and memory values for messages through the memory function to realize memorable communication.

The paper theoretically analyzes and explains that the model has the characteristics of traceability, synchronization, and batch signature authentication. The communication parties accumulate the status of each sent and received message in sequence into the memory value. Message traceability can be realized through message-tracing evidence and memory values. Message synchronization verification can be achieved by comparing memory values. Additionally, batch signature authentication can be implemented based on memory values. Through a single signature authentication, the integrity and authenticity of multiple messages are ensured, greatly reducing the overhead of signature authentication.

Through comparative analysis, this paper explores the transmission efficiency, traceability efficiency, and security performance of memorable communication.

2. Memory Communication Method

In this section, we elaborate on the design of a memorable communication model based on cryptographic accumulators. This model consists of a closed-loop system with the sender, receiver, and transmission channel, where each entity performs different functions to ensure communication security. On this basis, the memorable communication process is constructed, including the sender's message sending and the receiver's message receiving processes.

In addition, we present two methods to construct the memory function MF: one based on the RSA cryptographic accumulator, relying on number theoretic assumptions, and one based on message hash chains, adopting iterative hashing. Both methods can build efficient and reliable MFs to suit the requirements of different application scenarios. Meanwhile, it is possible to construct a key hash chain using memory values, allowing for the iterative update of encryption keys during the data transmission process.

2.1. Notation and Meaning

The following gives definitions and descriptions of some notations, as shown in the Table 1:

Table 1. Symbol explanation in the memorable communication method.

Symbol	Explanation
m_i	The i^{th} message of message transmission sequence
c_i	The ciphertext corresponding to the i^{th} message
x_i	The message element corresponding to m_i
MF	The memory function
Mem	The memory value
$prekey$	The pre-shared key between the communicating parties
w_i	The message-tracing evidence corresponding to m_i
X	Set of message elements, X = $\{x_1, ..., x_n\}$
MEM	Set of memory values, MEM = $\{Mem_1, ..., Mem_n\}$

Table 1. *Cont.*

Symbol	Explanation
K_i	The key corresponding to the m_i
seq	Sequence number of the message
χ	The value range of the message element
$f_{pr}(.)$	The prime transpose operation
$H(.)$	The cryptographic hash function
$A\|\|B$	Concatenate string A with string B

2.2. Memorable Communication Model

The memorable communication model is mainly composed of three subjects, including sender, receiver, and channel. The overall architecture is shown in Figure 1.

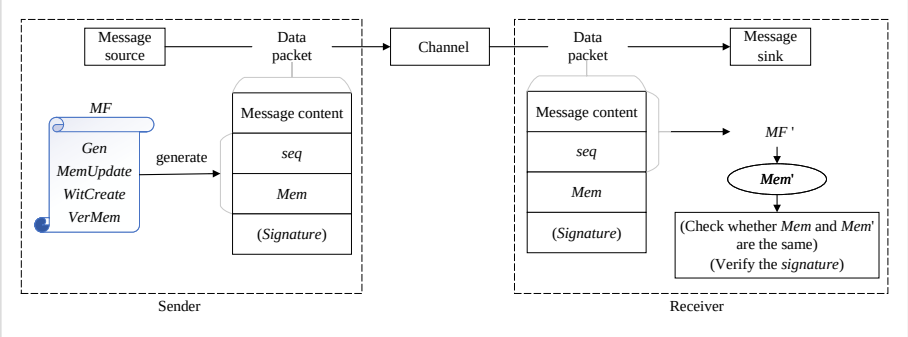

Figure 1. Overall architecture diagram.

(1) Sender: Before sending a message, the sender negotiates the memory function and related parameters with the receiver and generates the initial memory value. Each time a message is sent, the sequence number of the message is generated, the message element is generated based on the message and the sequence number, and the message element is added to the memory value. Finally, the sequence number and the memory value of the message are appended to the message and sent to the receiver. If necessary, you can also sign the message and attach the signature value to the message. When a message needs to be traced, the sender can generate corresponding message-tracing evidence. The trace can verify the existence of the message based on the message-tracing evidence.

(2) Receiver: Before receiving a message, the receiver generates a local initial memory value based on the memory function negotiated with the sender. When receiving a message, update the local memory value based on the message and its sequence number. If synchronization authentication is required, you can check whether message synchronization is successful by comparing the local memory value with the memory value in the message. If signature authentication is required, you can verify the messages with signatures. When a message needs to be traced, the receiver can also generate corresponding message-tracing evidence. The trace can verify the existence of the message based on the message tracing evidence.

(3) Channel: The transmission channel through which data are transmitted from the sender to the receiver.

2.3. Memory Function

Through the design of the memory function, the memorable communication model realizes endogenous security protection for communication, which lays the foundation for subsequent security analysis and proof. Regarding the memory function MF, we give the following definition:

Definition 1. *A general memory function can be called MF, MF = (Gen, MemUpdate, WitCreate, VerMem).*

- *Gen: Represents the algorithm for generating initial parameters and the initial memory value. By inputting security parameters, initial parameters and the initial memory value (which can be empty) are generated.*
- *MemUpdate: Represents the memory value update algorithm. The sender or receiver updates according to m, first converts m to x, then adds x to Mem, and finally updates* $Mem_{X'} = Mem_{X\cup\{x\}}$.
- *WitCreate: The sender or receiver creates w for m, first converts m to x, and then generates the corresponding w according to x. The type of w is divided into member evidence and non-member evidence.*
- *VerMem: Represents the message-tracing algorithm. The sender or receiver verifies the existence of m, first converts m to x, and then verifies the existence of the message according to x, Mem, and w. The authentication mode can be divided into member authentication and non-member authentication according to the type of w. If the verification result is 1, it means that m is indeed sent or received by w's provider, and 0 means that m is not sent or received by w's provider.*

2.4. Two Methods of Constructing Memory Function

In this section, we will present formal descriptions of the concrete memory functions based on the RSA accumulator and the message hash chain.

2.4.1. A Memory Function Based on RSA Accumulator

RSA accumulator can prove whether elements exist in the set efficiently. Here, a memory function is constructed based on an RSA accumulator, which can realize efficient message tracing with a small storage cost. The specific structure of the memory function based on the RSA accumulator is as follows:

- $Gen(1^\lambda)$: The sender or receiver inputs the security parameter λ to generate the key pair and the initial memory value. The key pair is $(sk, pk) = ((p, q), N)$, the initial memory value is $Mem_0 = g \in Z_n$, and N consists of two large strong prime numbers p and q, and $N = pq$.
- $MemUpdate(Mem_X, m, X, pk)$: The sender or receiver adds m to Mem_X, first calculating $x = f_{pr}(m||seq)$, $x \in \chi$ and then updating $Mem_{X'} = Mem_{X\cup\{x\}} = {Mem_X}^x mod N$.
- $WitCreate(Mem_X, m_i, X, pk)$: The sender or receiver generates message-tracing evidence for m_i, first calculating $x_i = f_{pr}(m_i||seq_i)$ and then generating $w_i = g^{u/x_i}\ modN$ (u=$\prod_{i=1}^{n} x_i$).
- $VerMem(Mem_X, m_i, w_i, pk)$: The sender or receiver verifies the existence of m_i by first calculating $x_i = f_{pr}(m_i||seq_i)$ and then judging whether $Mem_X = {w_i}^{x_i} mod N$ is true. If true, it outputs 1; if not, it outputs 0. An output of 1 indicates that m_i was indeed sent or received, and 0 indicates that m_i was not.

2.4.2. A Memory Function Based on Message Hash Chain

The main idea of a message hash chain is to iteratively hash the hash value of the transmitted message to form a hash chain about the message sequence. Based on it, a memory function is designed here, which is constructed as follows:

- $Gen(1^\lambda)$: The sender or receiver input the security parameter λ to generate Mem_0. Mem_0 can be null.
- $MemUpdate(Mem_X, m, X)$: The sender or receiver adds m to Mem_X, first calculating $x = H(m||seq)$, $x \in \chi \backslash X$ and then updating $Mem_{X'} = Mem_{X\cup\{x\}} = H(Mem_X||x)$.
- $WitCreate(m_i, X, MEM)$: The sender or receiver generates message-tracing evidence for m_i, first calculating $x_i = H(m_i||seq_i)$ and then generating $w_i = (w_a, w_b)$, $w_a = [Mem_0, Mem_d, Mem_{2d}, ..., Mem_n]$, $w_b = [x_{j*d}, ..., x_{i-1}, x_{i+1}, ..., x_{(j+1)*d}]$, d is the in-

terval of evidence nodes, $d = \sqrt{n}$, $j*d < i \leq (j+1)*d$, w_a is the set of evidence nodes, and w_b is the set of message elements between the two evidence nodes closest to x_i.

- $VerMem(Mem_X, m_i, w_i)$: The sender or receiver verifies the existence of m_i. First, determine whether $Mem_X \in w_a$ is true. If not, output 0. If true, calculate $x_i = H(m_i||seq_i)$, judge whether $Mem_{(j+1)*d} = H(Mem_{j*d}||...||x_i||...||x_{(j+1)*d})$ is true, if true, output 1, if not, output 0. An output of 1 indicates that m_i was indeed sent or received, and 0 indicates that m_i was not.

2.5. Chain-Key Module

This section introduces the process of updating encryption keys for message encryption during communication. This process can utilize a hash algorithm to generate a key hash chain, therefore achieving the encryption of various messages using different keys.

- $Gen(1^\iota)$: For the input security parameter ι, using a key negotiation algorithm, negotiate and derive the initial key k_1.
- $Enc(m_i, k_i)$: Encrypt the message m_i using the key k_i to obtain the ciphertext c_i.
- $Dec(c_i, k_i)$: Decrypt the ciphertext c_i using the key k_i to obtain the plaintext m_i.
- $KeyUpdate(mem_{x_i}, k_i)$: After the sender or receiver updates the memory value mem_{x_i}, the next key $k_{i+1} = H(Mem_{x_i}||k_i)(i > 1)$ is updated.

2.6. Communication Process

In this method, the specific processes of sending, receiving, and tracing messages are different. This section will describe the specific steps of sending and receiving messages. First, the packet format for this type of data packet during transmission is shown in Figure 2 below. The network layer and transport layer packet formats are consistent with the TCP/IP protocol. The application-layer data is sent in TLV (Type, Length, Value) format, where Type represents the meaning of the field, Length represents the length of the field, and Value represents the value of the field. The important information such as the sequence number, memory value, signature, and data are determined according to the meaning represented by Type.

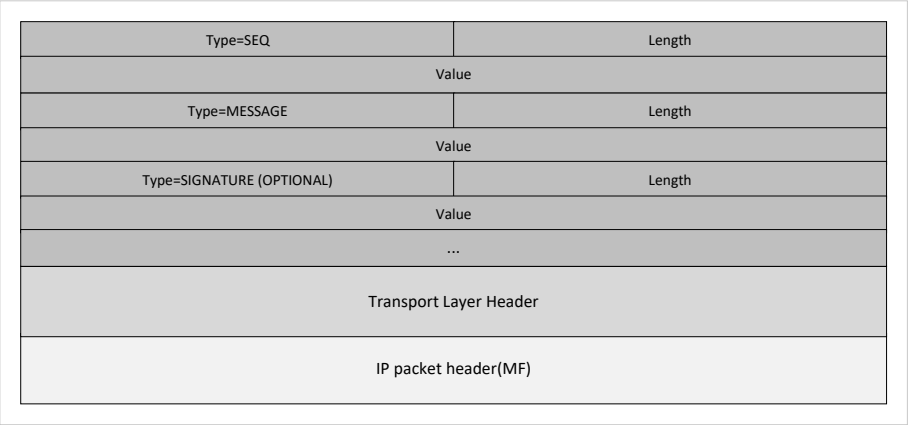

Figure 2. Packet Format with Memory Value.

2.6.1. Message Sending Process

The procedure for sending messages to the sender is as follows:

Step 1: The initial parameters and Mem_0 are generated according to the *Gen* in the memory function. If synchronization is needed, the same memory function can be negotiated with the receiver. If signature authentication is required, negotiate the signature interval with the receiver;

Step 2: Generate the corresponding *seq* for the *m* to be sent;

Step 3: Generate *x* for the *m*;

Step 4: Generate corresponding $Mem_{X'}$ for the m according to $MemUpdate$ in the memory function;

Step 5: If necessary, sign the m and its corresponding $Mem_{X'}$, and encrypting the message m using the key k_i and updating the key to k_{i+1};

Step 6: The authentication data (some contain only seq and $Mem_{X'}$, some contain seq, $Mem_{X'}$ and signature of the m) is attached to the m;

Step 7: Send this message to the receiver.

The specific process diagram of the sender is shown in Figure 3.

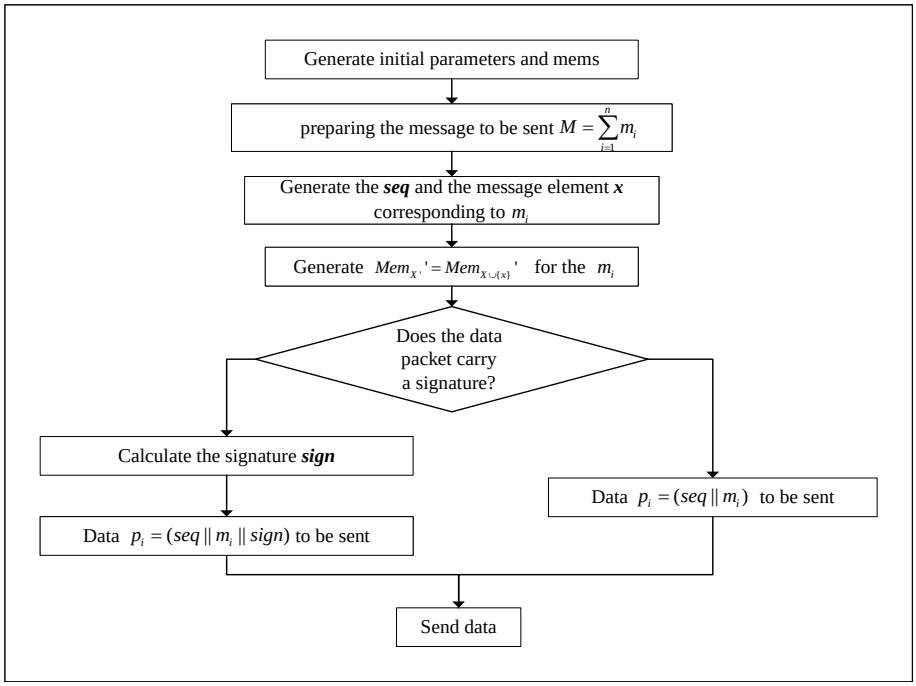

Figure 3. Sender Process Diagram.

2.6.2. Message Receiing Process

The procedure for receiving mvessages from the receiver is as follows:

Step 1: The initial parameters and Mem_0 are generated according to the Gen in the memory function. If synchronization is needed, the same memory function can be negotiated with the sender. If signature authentication is required, negotiate the signature interval with the sender;

Step 2: Receive messages sent by the sender (If synchronization is required, the messages are sequentially arranged according to seq, and subsequent operations are performed. If there is no synchronization requirement, directly receive the messages). Messages are divided into messages that carry only seq and memory value and messages that carry seq, memory value, and signature.

Step 3: Generate x' for the m;

Step 4: Generate corresponding $Mem'_{X'}$ for the m according to $MemUpdate$ in the memory function;

Step 5:If there is a synchronization requirement, check whether $Mem_{X'}$ in the received message is consistent with $Mem'_{X'}$ constructed locally. If they are consistent, the synchronization succeeds. If they are inconsistent, the synchronization fails, and an error is reported;

Step 6: If the message carries a signature, the corresponding public key can be used to verify the signature. If the verification fails, the message is discarded, an error is reported, and the message is decrypted that needs to be decrypted using the key k_i and updating the key to k_{i+1}.

The specific process diagram of the receiver is shown in Figure 4.

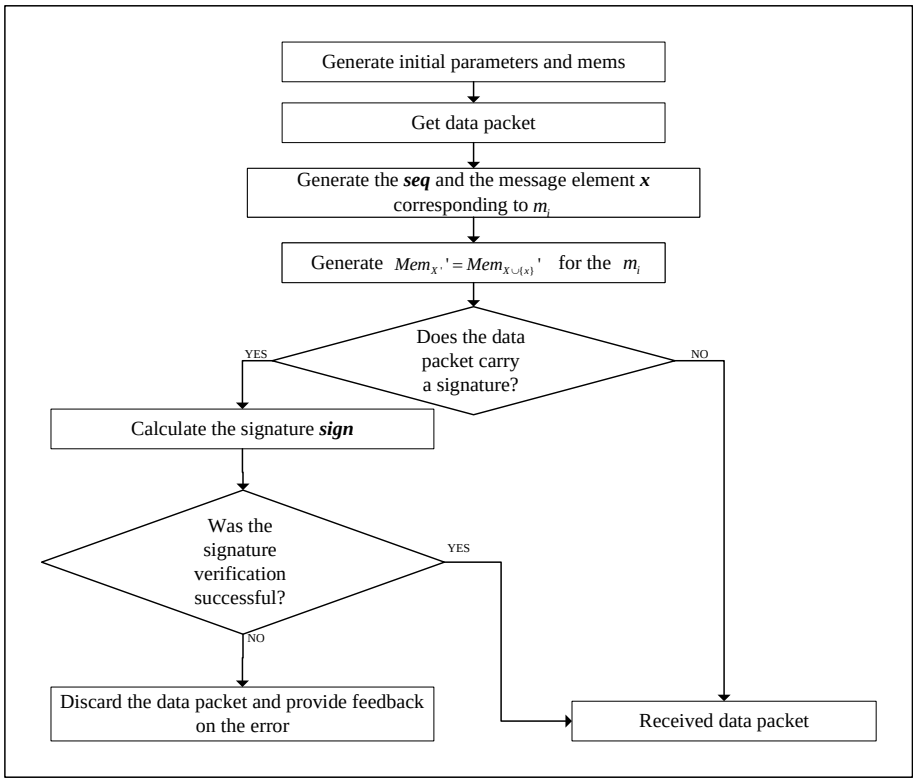

Figure 4. Receiver Process Diagram.

2.7. Message-Tracing Process

The specific steps of message tracing are as follows:

Step 1: The sender or receiver initiates a traceability request against a certain m;

Step 2: The sender or receiver generates x for the m;

Step 3: The sender or receiver generates the corresponding w for the m according to the *WitCreate* in the memory function;

Step 4: The sender or receiver verifies the existence of the m according to the *VerMem* in the memory function and w. If the verification value is 1, it indicates successful traceability, i.e., the m exists; if the verification value is 0, it indicates failed traceability, i.e., the m does not exist.

3. Model Analysis

The security threats facing network communication are complex and ever-changing. In this section, we choose to elaborate on the aspects of synchronicity, traceability, and anti-forgery. Through theoretical derivation and proof, we comprehensively validate the multiple security features of this model.

3.1. Synchronization

In this subsection, the synchronicity analysis of the memorable communication model is described through the following definitions and theoretical derivations. The synchronous nature of the memorable communication model is primarily demonstrated through logical reasoning and proof by contradiction. This establishes the foundation for the overall security attributes of the method.

Definition 2. *If for any k, there is always a* ϵ_0 *such that* $f(\epsilon) < \frac{1}{\epsilon^k}$ *when* $\epsilon > \epsilon_0$*. Then* $f(\epsilon)$ *is said to be a negligible value with* ϵ *as the parameter.*

Definition 3. Acc_{RSA} *is an RSA cryptographic accumulator. If it has the following three features,* Acc_{RSA} *is a good RSA cryptographic accumulator.*

(1) For a $f_{pr}(\cdot)$ in Acc_{RSA}, if it can find $a, b (a \neq b)$ in polynomial time such that the probability of $f_{pr}(a) = f_{pr}(b)$ is equal to $f(\epsilon)$, then the $f_{pr}(\cdot)$ in Acc_{RSA} is said to be non-collision.

(2) It is assumed that adversary A can arbitrarily select the set $X \subseteq M$(M is the range of accumulative values) of elements to be accumulated and initialize the accumulator. If A adds an arbitrary element x to X, A must generate corresponding evidence w_{op}, which can be verified by the evidence update checking algorithm CheckUpdate (output 1 means that the updated accumulated value and evidence are valid. An output of 0 indicates that the updated accumulative value and evidence are invalid. If the current accumulative value $\mathscr{A}cc_{bf} \Rightarrow X$ and $CheckUpdate(x, \mathscr{A}cc_{bf}, \mathscr{A}cc_{af}, w_{op}) = 1$, there is no updated accumulative value $\mathscr{A}cc_{af} \Rightarrow X \cup \{x\}$, Acc_{RSA} can safely add elements.

(3) Suppose that there is an adversary A that can arbitrarily generate its member evidence for elements in the cumulative element set $X \subseteq M$ and can also generate non-member evidence for elements outside X. If the evidence generated by A is w for an element x, when $\mathscr{A}cc \Rightarrow X$ and $x \in X$, the probability of $Belongs(x, w, \mathscr{A}cc) = 0$ is $f(\epsilon)$; When $\mathscr{A}cc \Rightarrow X$ and $x \notin X$, the probability of $Belongs(x, w, \mathscr{A}cc) = 1$ is $f(\epsilon)$, Acc_{RSA} is said to be undeniable in evidence calculation (Belongs represents the evidence verification algorithm, and if the output is 1, it means that the member proof is valid; If the output is 0, the non-member proof is valid. If the output is $\perp$, the evidence is invalid).

Definition 4. *Please note that H is a hash function. H is said to be a good hash function if it has the following two features.*

(1) Given a hash value h calculated using H, the H is said to be unidirectional if a particular input x can be found in polynomial time such that the probability of $h = H(x)$ is $f(\epsilon)$, and given any y, it is easy to calculate $H(y)$.

(2) For a H, if it can find $a, b, a \neq b$ in polynomial time such that the probability of $H(a) = H(b)$ is equal to $f(\epsilon)$, then the H is said to be non-collision.

Definition 5. *For the communication based on MF in this paper, the sender sends m_i in the message sequence $M = \{m_0, ..., m_n\}$ to the receiver one by one. When sending the message, the sender constructs Mem_i based on MF and attaches it to the corresponding m_i. The latest memory value constructed by the sender is recorded as Mem_n. The receiver constructs the local Mem'_i according to the received mi' and the local MF'. The message sequence received by the receiver is denoted as M', and the local latest memory value constructed is Mem'_n. If the receiver can ensure that the probability of $M \neq M'$ is $f(\epsilon)$ when $Mem_n = Mem'_n$ is obtained, then the communication method based on the MF in this paper has synchronization.*

Corollary 1. *MF_{RSA} is a memory function constructed based on RSA cryptographic accumulator. If the RSA cryptographic accumulator that constitutes MF_{RSA} is a good RSA cryptographic accumulator, then the communication method based on MF_{RSA} in this scheme has synchronization.*

Proof. Suppose the message sequence sent by the sender is $M = \{m_0, ..., m_n\}$, the sequence number is *seq*, the generated corresponding message element sequence is $X = \{x_0, ..., x_n\}$, and the generated memory value sequence is $MEM = \{Mem_0, ..., Mem_n\}$. The message sequence received by the receiver is $M' = \{m'_0, ..., m'_n\}$, the generated message element sequence is $X' = \{x'_0, ..., x'_n\}$, and the generated memory value sequence is $MEM' = \{Mem'_0, ..., Mem'_n\}(n \in N^*)$. Assuming that the communication method based on MF_{RSA} in this paper does not have synchronization, i.e., when the RSA cryptographic accumulator used in MF_{RSA} is a good RSA cryptographic accumulator and the communication parties communicate according to the method in this paper, the communication parties still cannot ensure that the data sent and received are completely consistent, i.e., $Mem_n = Mem'_n, M \neq M'$. Then there must be the following two situations:

(1) $Mem_i = {Mem_{i-1}}^{x_i} mod N = {Mem_{i-2}}^{x_{i-1} \cdot x_i} mod N = {Mem_{i-2}}^{x_i \cdot x_{i-1}} mod N$, when $Mem_n = Mem'_n$ alone, it cannot be determined that the sequence of message elements in X and X' is consistent, i.e., it cannot be determined that the sequence of M and M' is consistent. However, in this paper, the receiver calculates the *Mem* after sorting the

messages according to *seq*, so *seq* can ensure the message sequence without being changed. If *seq* is attacked and the attacker forges seq_i as $seq_i^*(seq_i \neq seq_i^*)$ and make $Mem_n = Mem_n'$, then $x_i = f_{pr}(m_i||seq_i) = f_{pr}(m_i||seq_i^*)$ must exist. However, this is contrary to the fact that the RSA cryptographic accumulator used in MF_{RSA} is a good RSA cryptographic accumulator, so this situation is not tenable.

(2) If the attacker directly forges m_i as $m_i^*(m_i \neq m_i^*)$ and makes $Mem_n = Mem_n'$, then $x_i = f_{pr}(m_i||seq_i) = f_{pr}(m_i^*||seq_i)$ must exist. However, this also contradicts that the RSA cryptographic accumulator used in MF_{RSA} is a good RSA cryptographic accumulator, so this situation is not tenable.

In conclusion, the original hypothesis is not valid, i.e., when the RSA cryptographic accumulator used in MF_{RSA} is a good RSA cryptographic accumulator, the memory value comparison between the communication parties can ensure the consistency of the message. Therefore, the communication method based on MF_{RSA} in this paper has synchronization. □

Corollary 2. *MF_{PHC} is a memory function constructed based on the message hash chain. If the hash function used in the message hash chain in this paper is good, then the communication method based on MF_{PHC} given in this paper has synchronization*

Proof. It is assumed that the communication method based on MF_{PHC} in this paper does not have synchronization, i.e., when the hash function used in MF_{PHC} is a good hash function when the communication parties follow the way in this paper, the communication parties still cannot guarantee the message consistency, i.e., $Mem_n = Mem_n', M \neq M'$. Since $Mem_i = H(Mem_{i-1}||x_i) = H(Mem_{i-2}||x_{i-1}||x_i) \neq H(Mem_{i-2}||x_i||x_{i-1})$, there is only the following situation. The attacker forges m_i as $m_i^*(m_i \neq m_i^*)$ and makes $Mem_n = Mem_n'$, then $H(m_i||seq_i) = H(m_i^*||seq_i)$ must exist, but this contradicts the hash function MF_{PHC} uses as a good hash function, so this situation is not valid. □

3.2. Traceability

This section presents a scheme for the traceability analysis of the memorable communication method. It mainly adopts logical reasoning and proof by contradiction to demonstrate that the memorable communication method has traceability, making ex-post facto network forensics possible.

Definition 6. *For the communication based on a memory function given in the paper, if the sender or receiver can determine whether a message has been sent or received based on the memory function, the communication method based on the memory function given in the paper is said to have traceability.*

Corollary 3. *MF_{RSA} is a memory function constructed based on RSA cryptographic accumulator. If the RSA cryptographic accumulator that constitutes MF_{RSA} is a good RSA cryptographic accumulator, then the communication method based on MF_{RSA} in this scheme has traceability.*

Proof. Assuming that the communication method of the base MF_{RSA} in this paper does not have traceability, i.e., the RSA cryptographic accumulator that constitutes the MF_{RSA} is a good RSA cryptographic accumulator, and both parties communicate according to the method given in this paper, both parties cannot correctly determine whether they have sent or received a certain message. Then there must be the following situation: the sender or receiver generates $w_i = WitCreate(Mem_X, m_i, X, p_k)$ for m_i, and the probability of $VerMem(Mem_X, m_i, w_i, pk) = 1$ is $f(\epsilon)$ in the case of the corresponding $x_i \in X$. However, this situation is inconsistent with the RSA cryptographic accumulator used in MF_{RSA} as a good RSA cryptographic accumulator, so this situation is not true. In conclusion, the original hypothesis is not valid, and the communication method based on MF_{RSA} given in this paper has traceability. □

Corollary 4. *MF_{PHC} is a memory function constructed based on the message hash chain. If the hash function used in the message hash chain in this paper is good, then the communication method based on MF_{PHC} given in this paper has traceability.*

Proof. Assuming that the communication method based on MF_{PHC} in this paper does not have traceability, i.e., when the hash function used in MF_{PHC} is good, and the communication parties communicate according to the way given in this paper, the communication parties cannot correctly determine whether a certain message has been sent or received. Then there must be the following situation: The sender or receiver generates $w_i = (w_a, w_b)$ for m_i, and $Mem_{(j+1)*d} = H(Mem_{j*d}||...||x_i||...||x_{(j+1)*d}) = H(Mem_{j*d}||...||x_i'||...||x_{(j+1)*d})(x_i \neq x_i')$ occurs when w_i is validated by $VerMem(Mem_X, m_i, w_i)$. However, this situation is inconsistent with the hash function used in MF_{PHC} as a good hash function, so this situation is not true. In conclusion, the original hypothesis is not tenable, and the communication method based on MF_{PHC} given in this paper has traceability. □

3.3. Batch Signature and Authentication

This subsection provides an analysis of batch signature authentication for the memorable communication model. Through the signing of memory values, the memorable communication model achieves efficient batch authentication, preventing threats such as man-in-the-middle attacks and ensuring communication security.

Definition 7. *For a digital signature scheme$(Genkey, Sign, Verify)$, Genkey represents the asymmetric key generation algorithm, privatekey is the private key of the signature, and publickey is the public key of the signature. Sign is the signature algorithm for a m, $Sign(privatekey, m) = s$, s is its signature result, and $Verify$ is the signature verification algorithm. For the s and the key pair $(privatekey, publickey)$ generated by Genkey, there is always $Verify(publickey, s) = 1$. The digital signature scheme is said to be secure if the probability of a correct signature is $f(\epsilon)$ only through publickey to forge privatekey in polynomial time.*

Definition 8. *For the communication method based on MF in this paper, the sender sends m_i in $M = \{m_0, ..., m_n\}$ to the receiver one by one according to the method in the paper, and the memory value corresponding to m_i is Mem_i. On this basis, if the sender adopts a digital signature scheme, uses privatekey in the key pair to generate $s_i = Sign(privatekey, Mem_i)$ for m_i, and sends m_i to the receiver after s_i is attached to the m_i. The receiver only uses publickey in the key pair to obtain $Verify(publickey, s_i) = 1$, which can confirm the authenticity, non-repudiation, and integrity of $m_0, ..., m_i$ in M. It can be said that the communication method based on MF in this paper has the feature of batch signature authentication.*

Corollary 5. *MF_{RSA} is a memory function constructed based on RSA cryptographic accumulator. Suppose the RSA cryptographic accumulator that constitutes MF_{RSA} is a good RSA cryptographic accumulator, and the signature scheme used in communication according to the method given in this paper is secure. In that case, the communication method with a signature based on MF_{RSA} in this scheme has the feature of batch signature and authentication.*

Proof. Set the sequence of the message sent by the sender as $M = \{m_0, ..., m_n\}$, the sequence of the message elements generated as $X = \{x_0, ..., x_n\}$, the memory value corresponding to m_i as Mem_i, and the result of the sender signing m_i as $s_i = Sign(privatekey, Mem_i)$ $(0 < i \leq n)$. Assuming that the communication method with signature based on MF_{RSA} in this paper does not have the feature of batch signature authentication: under the condition that the RSA cryptographic accumulator that constitutes the MF_{RSA} is a good RSA cryptographic accumulator and the digital signature scheme is secure, the receiver cannot confirm the authenticity, non-repudiation, and integrity of $m_0, ..., m_i$ in M only by calculating $Verify(publickey, s_i) = 1$ with $publickey$. There must be the following in two situations:

(1) The attacker tampers M to $M^* = \{m_0^*, ..., m_i^*, m_{i+1}, ..., m_n\}(m_i^* \neq m_i)$ and generates $X^* = \{x_0^*, ..., x_i^*, x_{i+1}, ..., x_n\}$, $x_i^* = f_{pr}(m_i^*||seq_i)$ so that $Mem_i^* = g^{x_0^* \cdot ... \cdot x_i^*} modN = g^{x_0 \cdot ... \cdot x_i} modN = Mem_i$ and $Verify(publickey, s_i) = 1$. However, the RSA cryptographic accumulator used in MF_{RSA} is a good RSA cryptographic accumulator, so this situation is not true.

(2) The attacker tampers M to $M^* = \{m_0^*, ..., m_i^*, m_{i+1}, ..., m_n\}(m_i^* \neq m_i)$ and generate $X^* = \{x_0^*, ..., x_i^*, x_{i+1}, ..., x_n\}$ so that $Mem_i^* = g^{x_0^* \cdot ... \cdot x_i^*} modN \neq Mem_i$, and forges $s_i^* = Sign(privatekey^*, Mem_i^*)$ so that $Verify(publickey, s_i^*) = 1$. However, this contradicts the premise that a digital signature scheme is secure, so this kind of situation is not tenable.

In summary, the hypothesis is not valid. In the form of the RSA cryptographic accumulator used in MF_{RSA} is a good RSA cryptographic accumulator, and the digital signature scheme is secure. The communication with signature based on MF_{RSA} in this paper has the feature of batch signature and authentication. □

Corollary 6. *MF_{PHC} is a memory function constructed based on the message hash chain. If the hash function used in the message hash chain in this paper is a good hash function and the signature scheme used in communication according to the method given in this paper is secure, then the communication method with signature based on MF_{PHC} in this scheme has the feature of batch signature and authentication.*

Proof. Assuming that the communication method with signature based on MF_{PHC} in this paper does not have the feature of batch signature authentication: under the condition that the hash function used in MF_{PHC} is a good hash function and the digital signature scheme is secure, the receiver cannot confirm the authenticity, non-repudiation, and integrity of $m_0, ..., m_i$ in M only by calculating $Verify(publickey, s_i) = 1$ with $publickey$. There must be the following two situations:

(1) The attacker tampers M to $M^* = \{m_0^*, ..., m_i^*, m_{i+1}, ..., m_n\}(m_i^* \neq m_i)$ and generates $X^* = \{x_0^*, ..., x_i^*, x_{i+1}, ..., x_n\}$ so that $Mem_i^* = H(x_0^*||...||x_i^*) = H(x_0||...||x_i) = Memi$ and $Verify(publickey, s_i) = 1$. However, as the hash function used in MF_{PHC} is a good hash function, this situation is not true.

(2) The attacker tampers M to $M^* = \{m_0^*, ..., m_i^*, m_{i+1}, ..., m_n\}(m_i^* \neq m_i)$ and generate $X^* = \{x_0^*, ..., x_i^*, x_{i+1}, ..., x_n\}$ so that $Mem_i^* = H(x_0^*||...||x_i^*) \neq Mem_i$, and forges $s_i^* = Sign(privatekey^*, Mem_i^*)$ so that $Verify(publickey, s_i^*) = 1$. However, this contradicts the premise that a digital signature scheme is secure, so this kind of situation is not tenable.

In summary, the hypothesis is not valid. Under the condition that the hash function used in MF_{PHC} is a good hash function and the digital signature scheme is secure, the communication method with signature based on MF_{PHC} in this paper has the feature of batch signature authentication. □

3.4. Randomness of Chain Keys

This section provides an analysis of the randomness among the keys generated based on the memory value in the chain-key generation. The analysis illustrates that the chain-key model based on strongly confused hashing possesses excellent random characteristics.

Data subjected to a strong mixing hash operation will be completely shuffled. The data before and after the strong mixing hash operation can be considered with a high probability of undergoing a random permutation, meaning non-randomness can be negligibly small.

Definition 9. *There exist two keys. If key B is obtained by some complex transformation from key A and key B cannot be deduced from key A, then keys A and B are said to have a derived relationship. The process of derivation is called a derivation algorithm. Key A is referred to as the master key, and key B is referred to as the derived subkey.*

For a master key A and a derived subkey B with a derived relationship, cracking key A can lead to obtaining key B. If key B cannot be obtained through any means other

than cracking key A, then the difficulty of cracking key B is equivalent to the difficulty of cracking key A.

Definition 10. *Computation Indistinguishability: Given two sequences* $\{X_n\}_{n\in N}$*and* $\{Y_n\}_{n\in N}$*, if there is no effective algorithm that can distinguish between them, then it is said that these two sequences are computationally indistinguishable.*

Definition 11. *Statistical Distance: Assuming two populations* $\{X_n\}_{n\in N}$*and* $\{Y_n\}_{n\in N}$*, represented by* $\Delta(n)$ *as the statistical distance between the two populations X and Y, the definition is as follows:*

$$\Delta(n) = \frac{1}{2}\sum_a |pr[X_n = a] - pr[Y_n = a]|$$

If $\Delta(n)$ *is negligibly small, it is said that* $\{X_n\}_{n\in N}$*and* $\{Y_n\}_{n\in N}$ *are statistically close. If they are statistically close, then they are indistinguishable.*

Based on the above theorems and definitions, the following two corollaries are derived:

Corollary 7. *The intermediate key* K_i *in the key chain and the initial key* K_1 *have a derived relationship. The difficulty for an attacker to obtain the intermediate key* K_i *will not be less than* $O(2^n)$*, meaning that it is not feasible for an attacker to have a higher probability of obtaining the intermediate key* K_i *when* K_1 *is unknown.*

Using proof by contradiction, the proof is conducted as follows:

Proof. Assume that an attacker can crack the intermediate key K_i with a higher probability when K_1 is unknown. Then, the attacker would need to acquire the correlation between at least two intermediate keys or more. Subsequently, employing analytical techniques such as differential analysis for key analysis would be required. Otherwise, this would contradict Shannon's proof of perfect security—one-time pad's absolute security. This implies that these two or more keys do not possess computational indistinguishability, meaning that statistically, $\Delta(n)$ is not negligible.

However, since the keys are computed through strong mixing hash operations, the conclusion that $\Delta(n)$ obtained in statistics is not negligibly small contradicts the conclusion of a random permutation with high probability as assumed by strong mixing hash operations. Therefore, the assumption is not valid, i.e., an attacker cannot crack the intermediate key K_i with a higher probability when K_1 is unknown. □

Corollary 8. *The security of keys in the key chain depends on the randomness of the hash function hash, the randomness of the plaintext sequence* $M = \{M_n\}_{n\in N}$*, and the confidentiality of the pre-shared key, independent of the intermediate key states.*

Using mathematical induction, the proof is conducted as follows:

Proof. (1) The initial key K_1 is obtained through the hash operation of the pre-shared key *prekey*. At this point, plaintext is not involved. Therefore, the security of the initial key depends on the randomness of the hash function and the confidentiality of the pre-shared key *prekey*. The generation of the second key $K_2 = H(mem_1||k_1)$ is jointly determined by the plaintext m_1, the key k_1, and the pseudo-random function H. Since K_1 is the result of a hash operation on the pre-shared key *prekey*, $k_2$satisfies Corollary.

(2) Assume that the key k satisfies the Corollary;

(3) For the key $K_{i+1} = H(mem_i||k_i)$, its security is jointly determined by the key K_i, the randomness of the hash function, the memory value of plaintext m_i. Since K_i satisfies Corollary, then K_{i+1} also satisfies Corollary. In other words, the security of K_{i+1} depends on the randomness of the hash function, the randomness of the plaintext sequence $M = \{m_n\}_{n\in N}$, and the confidentiality of the pre-shared key.

(4) From (1), (2), and (3), it can be concluded that the Corollary holds. □

3.5. Attacks Analysis

This section will analyze and explain common network communication attacks.

3.5.1. Man-In-The-Middle (MITM) Attacks

MITM (Man-in-the-Middle) attacks involve attackers secretly relaying and potentially altering communication between two parties who trust each other for direct communication. This model can resist MITM attacks due to the use of secure technologies such as cryptographic accumulators and batch signature authentication.

3.5.2. Spoofing Attacks

Deceptive attacks involve attackers impersonating other devices or users on the network to steal data, spread malware, or bypass access controls. This scheme mitigates this risk by:

The use of memory functions ensures that each participant in the communication has a verifiable and unique identity tied to their messages. This makes it much harder for an attacker to impersonate a legitimate user without access to their specific cryptographic materials.

The system's traceability feature, which allows the sender and receiver to verify the origin and integrity of messages, acts as a deterrent to spoofing. Any discrepancy in the traceability check would indicate a potential spoofing attempt.

3.5.3. Session Hijacking

Session hijacking involves taking over a user's session to gain unauthorized access to information or services. The proposed system counters this threat through:

Ensuring that messages are synchronized and cannot be repudiated helps in maintaining a secure and continuous session. Any attempt to hijack the session would be detected as an anomaly in the sequence of message exchanges, thanks to the cryptographic linkages provided by the memory functions.

By updating encryption keys per data packet based on the memory values, the system ensures that even if a session key is compromised, it cannot be used to hijack the session, as future communications will use different keys.

3.6. Quantum Resistance Analysis

Traditional information systems continue to confront threats from computer attacks, struggling to withstand cryptographic analyses and other attack methodologies executed by powerful computers. The emergence of quantum computing further exacerbates this threat.

The communication model designed in this paper, based on MF_{PHC}, primarily relies on maintaining a memory function to ensure the integrity and consistency of data transmission while enabling the traceability of messages. The core concept of this model is relatively independent of specific network types and is theoretically applicable to various network environments, whether they be classical or quantum in nature. Additionally, unlike public-key cryptography, traditional hash functions are considered to be capable of resisting quantum attacks by increasing the length of the existing hash output [20]. With the continuous development of quantum technology, numerous novel algorithms have emerged to counter quantum attacks, such as quantum digital signatures and certificates that prevent signature forgery [21,22], as well as new hash functions designed to resist quantum computing [23]. In a quantum environment, these interchangeable algorithms can autonomously negotiate as needed. The model possesses the capability to resist quantum computing.

Various methods are usually available for the distribution of pre-shared keys. In a quantum network environment, quantum key distribution (QKD) is commonly employed to secure key distribution and resist quantum attacks.

4. Comparison and Analysis

In this section, the memorable communication method is compared and analyzed from three aspects: communication efficiency, traceability efficiency, and security features. Through theoretical comparison and contrastive analysis, the advantages of the memorable communication method in terms of efficiency and security are fully demonstrated. It guarantees secure transmission and achieves efficient traceability while also possessing various security features such as synchronization and non-repudiation.

4.1. Comparative Analysis of Communication Efficiency

Table 2 analyzes and compares the efficiency of the basic communication method based on MF_{RSA}, basic communication method based on MF_{PHC}, communication method with signature based on MF_{RSA}, communication method with signature based on MF_{PHC} and ordinary packet-by-packet signature communication method. Assume that the time required for a hash operation is T_h, the time required for a prime transpose operation is T_P, the time required for an exponential operation is T_i, the time required for a signature and verification is T_s and T_v, respectively, and the signature interval of communication methods with signatures based on MF_{RSA} and MF_{PHC} is m. The theoretical time consumption of transmitting n messages by several communication methods is shown in Table 2.

As shown in Table 2, the time cost of the basic communication method based on MF_{PHC} is less than that of the basic communication method based on MF_{RSA} because $(T_p + T_i) > 2T_h$, and the sum of the time required by a prime transpose operation and an exponential operation is greater than two hash operations. In addition, the communication method with signature based on MF_{RSA} and MF_{PHC} has more hash operations, exponential operations, and prime transpose operations than packet-by-packet signature communication method, but signature and verification signature operations require much more time than these operations, and interval signature makes the communication methods with signatures based on MF_{RSA} and MF_{PHC} have lower time cost in general, and the time cost advantage depends on the size of m.

Table 2. Efficiency analysis of different communication methods.

Different Communication Methods	Time Consumption
Basic communication method based on MF_{RSA}	$2n(T_p + T_i)$
Basic communication method based on MF_{PHC}	$4nT_h$
Communication method with signature based on MF_{RSA}	$2n(T_p + T_i) + \lceil \frac{n}{m} \rceil (T_s + T_v)$
Communication method with signature based on MF_{PHC}	$4nT_h + \lceil \frac{n}{m} \rceil (T_s + T_v)$
Packet-by-packet signature communication method	$2nT_h + n(T_s + T_v)$

4.2. Comparative Analysis of Message-Tracing Efficiency

Table 3 analyzes and compares the message-tracing evidence generation efficiency, evidence verification efficiency, and storage cost of MF_{RSA} and MF_{PHC}, respectively. Assume that the time required for a hash operation is T_h, the time required for a prime transpose operation is T_p, the time required for an exponential operation is T_i, and the maximum length of a message is n. The traceability of MF_{RSA} and MF_{PHC} is shown in Table 3.

As shown in Table 3, the time consumption of evidence generation for MF_{RSA} is greater than that for MF_{PHC}. The time consumption of evidence verification of MF_{RSA} is much less than that of MF_{PHC}. The storage cost of MF_{RSA} is $u = \prod_{i=1}^{n} x_i$, which is the product of all message elements. The space complexity of MF_{RSA} is smaller than that of $O(n)$. The storage cost of MF_{PHC} is $MEM + X$, which is the set of message elements and memory values. The space complexity of MF_{PHC} is $O(2n)$. In general, the message traceability of MF_{RSA} is higher than that of MF_{PHC} because MF_{RSA} has great advantages in evidence verification and storage.

Table 3. Comparison of the traceability of the two memory functions.

Memory Function	Evidence Generation Time	Evidence Verification Time	Storage Cost
MF_{RSA}	$T_p + T_i$	$T_p + T_i$	u
MF_{PHC}	T_h	$(\sqrt{n}+1)T_h$	$MEM + X$

4.3. Comparison of Security Features

To illustrate the security properties of the communication method with signature based on the two memory functions in this paper, this paper compares it with the communication method based on IP protocol, the packet-by-packet signature communication method, the communication method based on AH protocol in IPsec and the communication method based on ESP protocol in IPsec. Differences in security attributes of different communication methods are shown in Table 4.

The communication method based on IP protocol is designed for transmission performance, but it lacks security attributes and is easy to cause various network security problems. The packet-by-packet signature communication can greatly improve the security of transmission but also greatly reduce the transmission efficiency. The communication method based on AH in IPsec ensures the integrity and non-tampering of transmission, and the communication method based on ESP ensures the confidentiality of message transmission. However, the two protocols in IPsec do not guarantee the non-repudiation of message transmission, nor do they have security features such as traceability and synchronization. The basic communication method based on MF_{RSA} and MF_{PHC} in this paper can guarantee the non-tampering and integrity of message transmission and realize the synchronization and traceability of the message. Based on MF_{RSA} and MF_{PHC}, messages can be signed and authenticated in batches. It ensures the integrity, authenticity, and non-repudiation of multiple messages by signing and authenticating them at a time, greatly reducing the cost of signature and authentication.

Table 4. Security comparison of different communication methods.

Communication Method	Non-Tampering	Integrity	Non-Repudiation	Traceability	Synchronization
IP protocol	No	No	No	No	No
Packet-by-packet signature	Yes	Yes	Yes	No	No
AH protocol	Yes	Yes	No	No	No
ESP protocol	Yes	Yes	No	No	No
Basic communication method based on MF_{RSA}	Yes	Yes	No	Yes	Yes
Basic communication method based on MF_{PHC}	Yes	Yes	No	Yes	Yes
Communication method with signature based on MF_{RSA}	Yes	Yes	Yes	Yes	Yes
Communication method with signature based on MF_{PHC}	Yes	Yes	Yes	Yes	Yes

5. Experimental Analysis

This section validates and analyzes the performance of the memorable communication model through experiments.

5.1. Experimental Environment

In this paper, the communication efficiency test experiment, evidence generation, and verification efficiency test experiment are all implemented by python computing language, and the programming tool is Pycharm 2021.1. The operating system used in the experiment was windows 10, and the PC was configured with an Intel (R) core (TM) i7-10875H CPU @ 2.30 GHz and 16 GB RAM.

The experimental code is deployed on multiple PCs and tested in a real network environment. This includes a sender and receiver for simulating data transmission. The experimental environment network topology is shown in Figure 5.

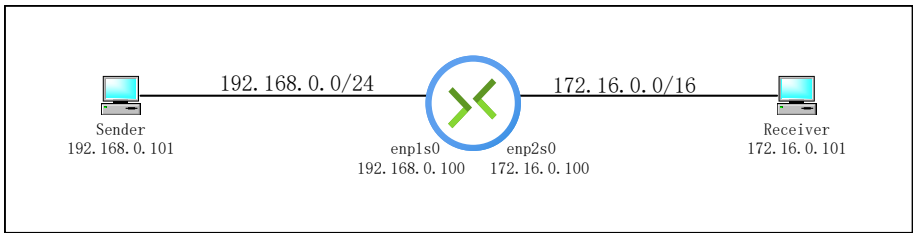

Figure 5. The experimental environment network topology.

5.2. Efficiency Testing of Communication

This paper tests the communication efficiency of the communication method based on IP protocol, the communication method based on MF_{PHC} and the communication method based on MF_{RSA}. The experiment tested the time consumed by different communication methods to transmit 1000 messages with lengths of 1024 bytes, 2048 bytes, 4096 bytes, and 8192 bytes. The experimental results are shown in Figure 6. Experimental results show that when the number of messages transmitted is the same, the communication time increases with the length of the message, which is positively correlated. The time consumed by the three communication methods is in the same order of magnitude, and the relationship is that the communication method based on MF_{RSA} is greater than the communication method based on MF_{PHC}, and the communication method based on MF_{PHC} is greater than the communication method based on IP protocol.

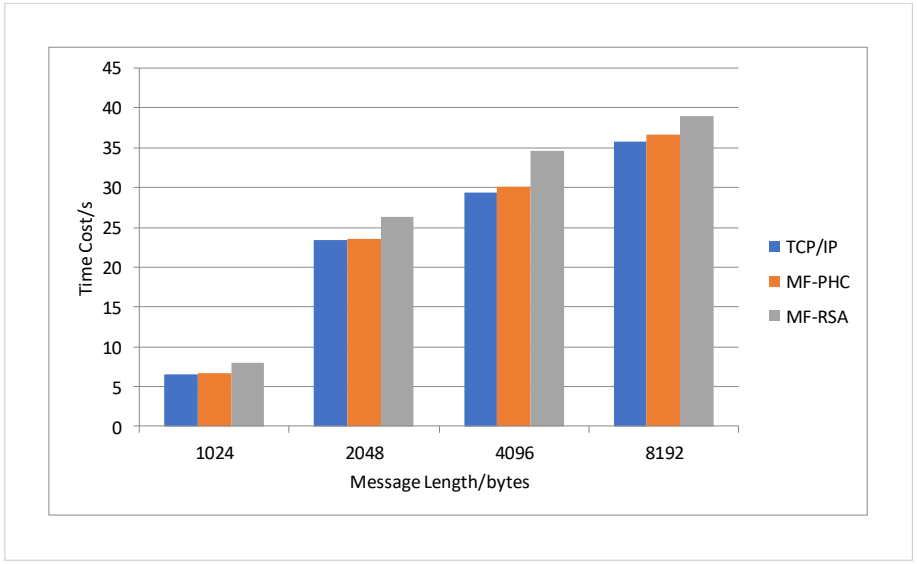

Figure 6. Communication time consumption under different message lengths.

The experiment also tested the time consumed by different communication methods to transmit 10, 50, 250, and 1000 messages of 8192 bytes. The experimental results are shown in Figure 7. The experimental results show that in the case of the same message length, the communication time increases with the number of messages, which is positively correlated. The time consumed by the three communication methods is in the same order of magnitude, and the relationship is that the communication method based on MF_{RSA} is greater than the communication method based on MF_{PHC}, and the communication method based on MF_{PHC} is greater than the communication method based on IP protocol.

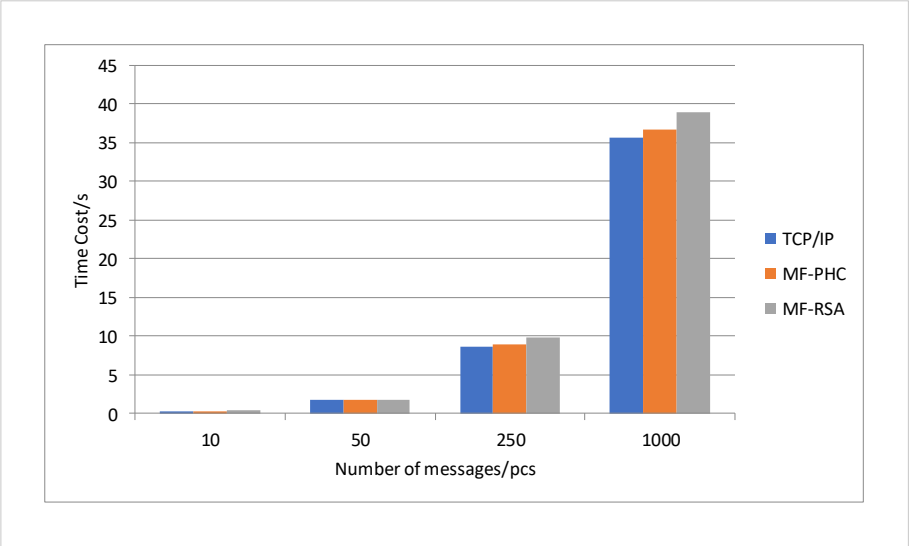

Figure 7. Communication time consumption under different message quantities.

5.3. Efficiency Testing of Evidence Generation and Verification

In this paper, we test the evidence generation and verification efficiency of MF_{RSA}. The experiments tested the time consumed to generate and verify evidence for messages with lengths of 1024 bytes, 2048 bytes, 4096 bytes, and 8192 bytes when the number of members was 100. The experimental results are shown in Figure 8. The experimental results show that when the number of members is the same, and the length of the message is different, the generation time and verification time of the evidence do not change significantly, and both fluctuate up and down. When the number of members is 100, the evidence generation time is much longer than the evidence verification time.

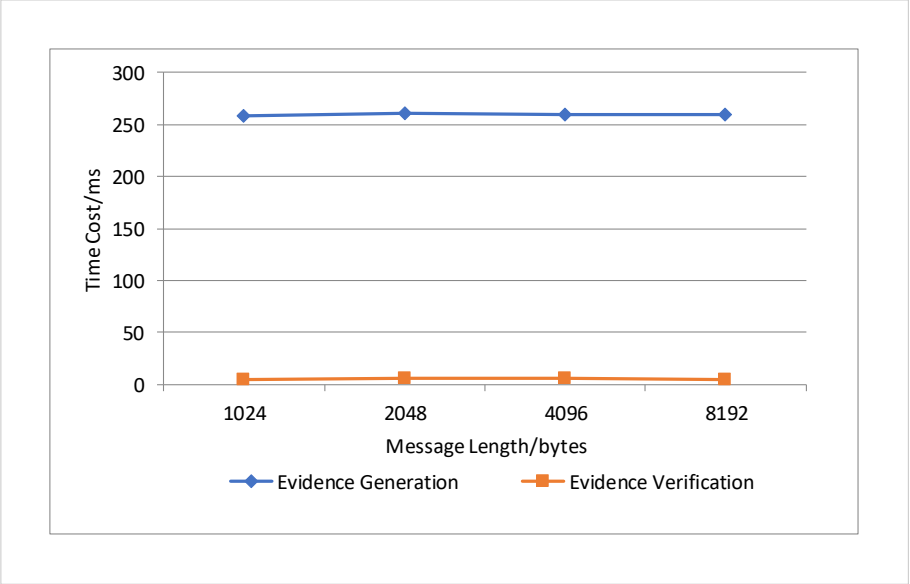

Figure 8. Evidence generation and verification time consumption under different message lengths.

The experiment also tested the time consumed to generate evidence and verify evidence for a message with a length of 8192 bytes when the number of members is 10, 50, 250, and 1000. The experimental results are shown in Figure 9. The experimental results show that when the message length is the same, the generation time of the evidence increases with the number of members, which is positively correlated, while the verification time of the evidence does not change significantly and fluctuates up and down. When the number

of members is 10, 50, 250, and 1000, the generation time of evidence is longer than the verification time of evidence.

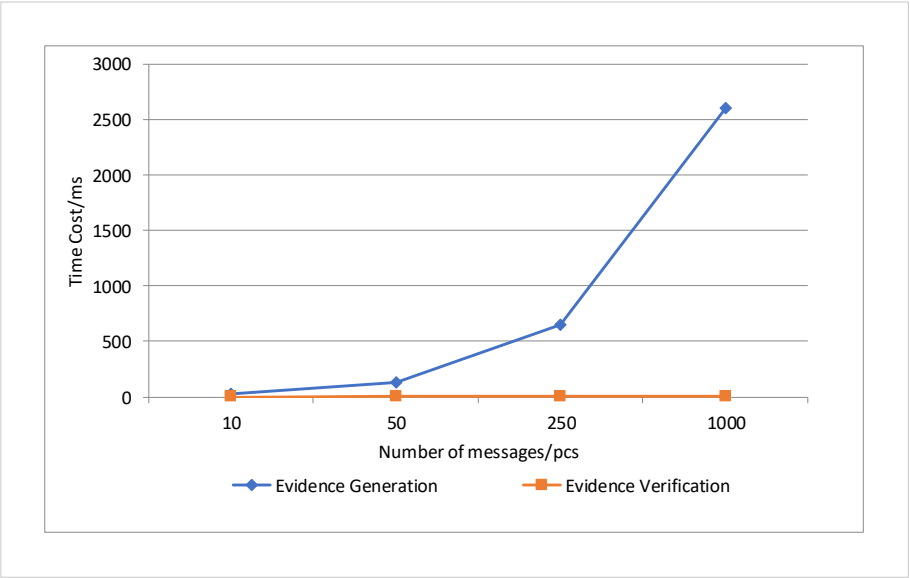

Figure 9. Evidence generation and verification time consumption under different message quantities.

5.4. Efficiency Testing of Batch Signature and Authentication

In this subsection, we tested the time taken for different signature intervals. The experiment involved testing the time required for signature intervals of packet-by-packet signing, signing in groups of 10, signing in groups of 20, and signing in groups of 100, with a member count of 1000. The experimental results are shown in Table 5. The experiments indicate that as the signature interval gradually increases, the time taken for signature generation and authentication decreases.

Table 5. Time Taken for Batch Signature Verification with Different Signature Intervals.

Signature Interval	Number of Signatures	Signature Generation Time	Signature Verification Time
packet-by-packet	1000	2583 ms	724 ms
groups of 10	100	232 ms	62 ms
groups of 20	20	46 ms	15 ms
groups of 100	10	24 ms	6 ms

However, the signature interval should not be excessively long, as it may result in some data packets not being authenticated for an extended period. It is recommended that the signature interval be dynamically adjusted based on network traffic conditions.

In addition, we conducted an experimental analysis of the communication performance of the theoretical analysis section. We signed every 100 packets and sent 1000 data packets. For signed A and B, we used 100 packets as intervals. The test results are shown in Table 6.

Table 6. Efficiency experiment testing for different communications methods.

Different Communication Methods	Time Consumption
Basic communication method based on MF_{RSA}	2603 ms
Basic communication method based on MF_{PHC}	480 ms
Communication method with signature based on MF_{RSA}	2715 ms
Communication method with signature based on MF_{PHC}	629 ms
Packet-by-packet signature communication method	3687 ms

Based on experimental data, it can be observed that batch signing exhibits greater efficiency. Compared to traditional per-packet signing, MF_{RSA} efficiency improves by approximately 40%. Due to the hash efficiency advantage of MF_{PHC}, the improvement in efficiency compared to traditional per-packet signing is approximately 450%.

5.5. Key Randomness Analysis

In the randomness analysis section, the evaluation will be conducted through two approaches. One is calculating the autocorrelation coefficients between generated keys, and the other is determining the overall probability and entropy of the generated keys. In relation to these two parameters, a comparison will be made between chained keys and IPsec. Within a single lifecycle (approximately 1 minute), the analysis will assess how the keys used for encrypting data undergo random variations for both scenarios.

5.5.1. Autocorrelation Coefficients

Autocorrelation coefficients are used to determine whether there is a correlation between values in a sequence. The values of autocorrelation coefficients range from -1 to 1. When the autocorrelation coefficient is close to -1, it indicates a strong negative correlation between the current value and the subsequent value. When the autocorrelation coefficient is close to 0, it indicates almost no correlation between the current value and the subsequent value. When the autocorrelation coefficient is close to 1, it indicates a strong positive correlation between the current value and the subsequent value.

To illustrate the randomness of keys generated by this scheme, the model will be used to generate keys with a length of 256 bits and a key generation period corresponding to an IPsec cycle (1 min). Within this cycle, the keys used for data encryption will be analyzed. Due to the lengthy nature of a 256-bit key, it is not conducive to analysis. Therefore, the 256-bit key will be split into 32 groups of 8-bit data. Autocorrelation coefficients will be calculated for each group of 8-bit data, resulting in 32 sets of autocorrelation coefficients.

According to the autocorrelation coefficient formula,

$$cor = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(x_{i+1} - \overline{x})}{\sum_{i=1}^{n}(x_i - \overline{x})^2}$$

Calculation of the autocorrelation coefficients for each group of 8-bit data can be performed using the autocorrelation coefficient formula. These coefficients represent the correlation between each byte of the 256-bit key. The specific results are illustrated in Figure 10 below:

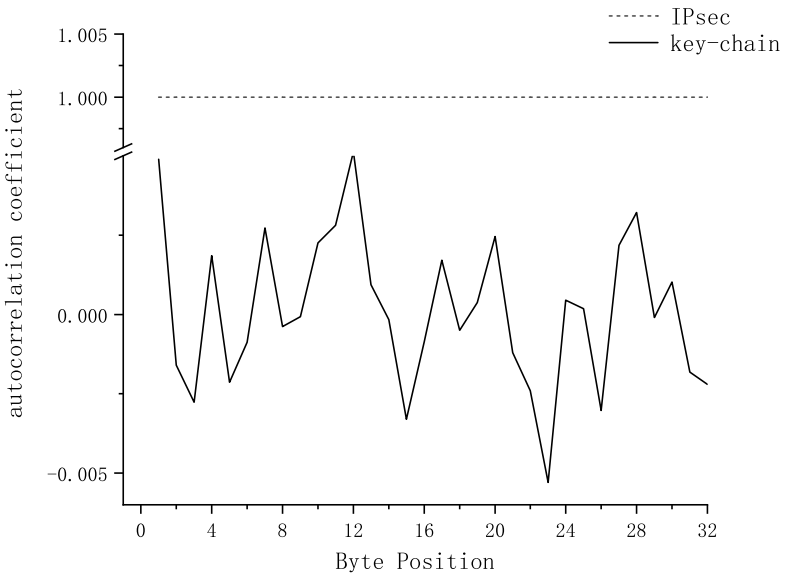

Figure 10. Autocorrelation Coefficient Analysis of chain-key and IPsec in One Cycle.

From this, it can be observed that within the same IPsec cycle, the correlation coefficients of chained keys are essentially stable within the range $[-0.005, 0.005]$. This indicates that the correlation between each byte of these keys is very low, with almost no linear correlation. In contrast, for IPsec, where the key is not updated, the correlation coefficient remains at 1.0.

5.5.2. Entropy

Entropy is a concept used to measure the uncertainty of a random variable and is typically employed to assess the randomness of a sequence. Higher entropy implies better randomness, while lower entropy suggests that the values of the random variable are more easily determined. For a discrete sequence like chained keys, the definition of entropy is as follows:

$$H(X) = -\sum_i p(x_i) log_2 p(x_i)$$

An explanation will be provided for the keys used for data encryption within a single IPsec cycle (1 min). Similarly, keys with a length of 256 bits will be divided into 32 groups of 8-bit bytes for independent analysis of each byte. The test analysis results for the entropy of each part are shown in Figure 11:

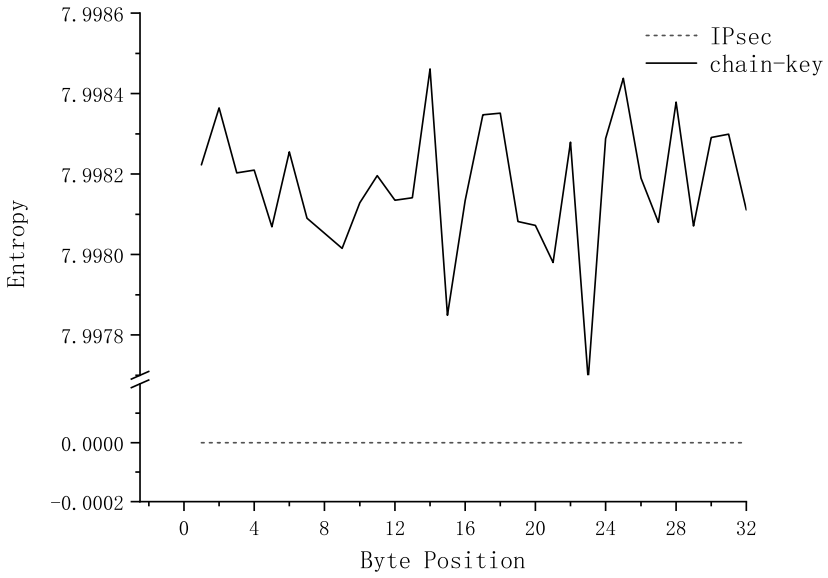

Figure 11. Autocorrelation Coefficient Analysis of chain-key and IPsec in One Cycle.

For uniformly distributed data with values ranging from 0 to 255, its entropy value $H_{max} = log_2(n) = log_2(256) = 8$. As shown in the Figure 11, the entropy value is approximately 7.9980, indicating that there is very little correlation between values in the sequence. The sequence tends to be highly random or uniformly distributed. In terms of randomness, this can be considered a positive feature. Overall, the entropy value of the chained keys suggests that the sequence exhibits characteristics very close to a uniform distribution and randomness in information theory. In contrast, due to the lack of key updates, the entropy value for IPsec is 0, indicating that the key remains fixed within a single lifecycle.

5.6. Experimental Conclusions

In this section, we quantitatively validated the efficiency of the aforementioned memorable communication method through experimental studies.

First, we compared the communication time overhead of methods based on MF_{PHC}, MF_{RSA}, and TCP/IP protocols. The results indicated that the time consumption for all three methods was of a similar order of magnitude, confirming the correctness of the

theoretical analysis. Additionally, the increase in the number and length of transmitted messages resulted in a linear increase in communication time, consistent with the analysis.

Second, we tested the evidence generation and verification time of MF_{RSA} under different message lengths and quantities. The results demonstrated that evidence generation time exceeded verification time, and evidence generation was independent of message length but positively correlated with the number of messages. This suggests the method's efficient traceability.

Next, we examined the time required for batch signature verification at different signature intervals. The results showed that longer signature intervals consumed less time. However, the choice of batch signature intervals should consider network communication load, as excessively large intervals may result in some packets not being signed and verified for an extended period.

Finally, we evaluated the randomness of the chain keys based on memory values, and the results showed that the chain keys exhibit good randomness.

From the experiments in this section, we can conclude that the memorable communication method indeed possesses the anticipated advantages in terms of communication efficiency and traceability. This provides a solid foundation for practical applications. Future work may involve conducting performance tests under larger-scale conditions to obtain a more comprehensive assessment.

6. Conclusions

This paper proposes a memorable communication method based on cryptographic accumulators. In this method, both communicating parties can verify the message data sent and received arbitrarily by the memory value, and the strong consistency of all message data can be guaranteed simply by comparing the memory value. This paper demonstrates the synchronization, traceability, batch signature, and authentication characteristics of the memorable communication method through theoretical proofs. Comparative analysis shows that the memorable communication method with a signature has more security advantages than IPsec and more efficiency advantages than a packet-by-packet signature. The memorable communication method with signature can not only guarantee the non-tampering and integrity of the transmission but also guarantee the non-repudiation, traceability, and synchronization of the transmission. Based on the memorable communication method, batch signature, and authentication can be realized, and the security of multiple messages can be guaranteed through one signature, which greatly improves the security transmission efficiency. Based on MF_{RSA}, batch signature is about 40% more efficient than signing each packet individually, while based on MF_{PHC}, the efficiency improvement is approximately 450% due to the advantages brought by hash operations and batch signature. This paper also demonstrates the communication efficiency of the memorable communication method based on MF_{PHC} and MF_{RSA} and the evidence generation and verification efficiency of MF_{RSA} through experimental analysis. The memorable communication method can be used as a network protocol in the Internet of Things, mobile communication, and other fields. How to improve the efficiency of message tracing further can be used as the next stage of research work.

Author Contributions: Conceptualization, W.J.; Methodology, W.J. and Y.W.; Software, S.Y.; Formal analysis, W.J.; Writing – original draft, Y.W. and S.Y.; Writing – review & editing, W.J., Y.W. and S.Y.; Project administration, W.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Key Research and Development Program of China, grant number 2022YFB2703000.

Data Availability Statement: The data presented in this study are available in this article.

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

References

1. Xu, K.; Fu, S.T.; Li, Q. The research progress on intrinsic internet security architecture. *Chin. J. Comput.* **2021**, *44*, 2149–2172.
2. Moskowitz, R.; Jokela, P.; Henderson, T.; Nikander, P. *Host Identity Protocol*; RFC 5012; Internet Engineering Task Force: Fremont, CA, USA, 2008.
3. Naylor, D.; Mukerjee, M.K.; Steenkiste, P. Balancing accountability and privacy in the network. *ACM SIGCOMM Comput. Commun. Rev.* **2014**, *44*, 75–86. [CrossRef]
4. Andersen, D.G.; Balakrishnan, H.; Feamster, N.; Koponen, T.; Moon, D.; Shenker, S. Accountable internet protocol (AIP). In Proceedings of the ACM SIGCOMM 2008 Conference on Data Communication, Seattle, WA, USA, 17–22 August 2008; pp. 339–350.
5. Frankel, S.; Krishnan, S. *IP Security (IPSEC) and Internet Key Exchange (ike) Document Roadmap*; 6071; Internet Engineering Task Force: Fremont, CA, USA, 2011.
6. Benaloh, J.; Mare, M.D. One-way accumulators: A decentralized alternative to digital signatures. In Proceedings of the Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, 23–27 May 1993; pp. 274–285.
7. Camenisch, J.; Lysyanskaya, A. Dynamic accumulators and application to efficient revocation of anonymous credentials. In *Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 61–76.
8. Cramer, R.; Gennaro, R. A secure and optimally efficient multiauthority election scheme. *Eur. Trans. Telecommun.* **1997**, *8*, 481–490. [CrossRef]
9. Li, J.T.; Li, N.H.; Xue, R. Universal accumulators with efficient nonmembership proofs. In Proceedings of the International Conference on Applied Cryptography and Network Security, Zhuhai, China, 5–8 June 2007; Springer: Berlin/Heidelberg, Germany, 2007; pp. 253–269.
10. Barić, N.; Pfitzmann, B. Collision-free accumulators and fail-stop signature schemes without trees. In Proceedings of the International Conference on the Theory and Applications of Cryptographic Techniques, Konstanz, Germany, 11–15 May 1997; Springer: Berlin/Heidelberg, Germany, 1997; pp. 480–494.
11. Au, M.H.; Tsang, P.P.; Susilo, W.; Mu, Y. Dynamic universal accumulators for ddh groups and their application to attribute-based anonymous credential systems. In Proceedings of the Cryptographers' Track at the RSA Conference, San Francisco, CA, USA, 20–24 April 2009; Springer: Berlin/Heidelberg, Germany, 2009; pp. 295–308.
12. Camenisch, J.; Kohlweiss, M.; Soriente, C. An accumulator based on bilinear maps and efficient revocation for anonymous credentials. In Proceedings of the International Workshop on Public Key Cryptography, Irvine, CA, USA, 18–20 March 2009; Springer: Berlin/Heidelberg, Germany, 2009; pp. 481–500.
13. Damga, R.D.I.; Triandopoulos, N. Supporting non-membership proofs with bilinear-map accumulators. *Cryptol. ePrint Arch.* **2008**.
14. Nguyen, L. Accumulators from bilinear pairings and applications. In Proceedings of the Cryptographers' Track at the RSA Conference, San Francisco, CA, USA, 14–18 February 2005; Springer: Berlin/Heidelberg, Germany, 2005; pp. 275–292.
15. Camacho, P.; Hevia, A.; Kiwi, M.; Opazo, R. Strong accumulators from collision-resistant hashing. In Proceedings of the International Conference on Information Security, Taipei, Taiwan, 15–18 September 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 471–486.
16. Lamport, L. Password authentication with insecure communication. *Commun. ACM* **1981**, *24*, 770–772. [CrossRef]
17. Golle, P.; Modadugu, N. Authenticating streamed data in the presence of random packet loss. In Proceedings of the Network and Distributed System Security Symposium, San Diego, CA, USA, 8–9 February 2001; pp. 13–22.
18. Han, M.X.; Jiang, W.B.; Guo, Y.N. Signature and authentication method based on message hash chain. *Appl. Res. Comput.* **2022**, *39*, 1183–1189.
19. Han, M.; Jiang, W. A Secure Communication Method Based on Message Hash Chain. *Appl. Sci.* **2022**, *12*, 4505. [CrossRef]
20. 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]
21. Bennett, C.H.; Brassard, G. Quantum cryptography: Public key distribution and coin tossing. *Theor. Comput. Sci.* **2014**, *560*, 7–11. [CrossRef]
22. Wang, L.J.; Zhang, K.Y.; Wang, J.Y.; Cheng, J.; Yang, Y.H.; Tang, S.B.; Yan, D.; Tang, Y.L.; Liu, Z.; Yu, Y.; et al. Experimental authentication of quantum key distribution with post-quantum cryptography. *NPJ Quantum Inf.* **2021**, *7*, 67. [CrossRef]
23. Krendelev, S.; Sazonova, P. Parametric hash function resistant to attack by quantum computer. In Proceedings of the 2018 Federated Conference on Computer Science and Information Systems (FedCSIS), Poznań, Poland, 9–12 September 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 387–390.