Journal of Network Intelligence ©2024 ISSN 2414-8105 (Online)
Taiwan Ubiquitous Information Volume **9**, Number **2**, May **2024**

Application of Proxy Signature Scheme Based on Blockchain in Multi Cloud Storage

Jian-Neng Chen*

Minnan Normal University
Zhangzhou, 363000, China
cjn610@163.com

Jia-Dong Wang

Minnan Normal University
Zhangzhou, 363000, China
1443645334@qq.com

Ke-Hao Tao

Minnan Normal University
Zhangzhou, 363000, China
760253221@qq.com

Yu-Ping Zhou

Key Laboratory of Data Science and Intelligence Application
Zhangzhou, 363000, China
yp_zhou@mnnu.edu.cn

Hong-Yi Li

Yeaosound Network Tech Ltd
Zhangzhou, 363000, China
lihongyi@yeaosound.cn

*Corresponding author: Jian-Neng Chen

Received May 10, 2023, revised September 17, 2023, accepted December 7, 2023.

Abstract. *As one of the most suitable online storage methods, cloud storage service is widely used by users with its convenient and efficient features. However, the existing cloud storage has some serious problems, mainly because it is difficult to manage the security and privacy of its database, and data storage cannot meet the security requirements of users. To increase security, blockchain technology is applied to data storage and data connectivity, which is reflected in the blockchain-based data storage model among multiple cloud providers. However, due to the limited storage space of blockchain transactions and the current speed of processing blockchain storage transactions, this model still has shortcomings. To resolve these issues, this paper proposes a proxy aggregation signature scheme to improve signature verification efficiency while compressing disk space and reducing communication bandwidth. The experimental results show that safe and efficient data storage can be achieved with this scheme.*

Keywords: Blockchain; Cloud storage; Proxy signature

1. **Introduction.**

With the rise of the Internet of Things [1], more and more information-aware devices are connected to the Internet, connecting people, devices, and things [2]. According to IDC data, there will be 41.6 billion IoT devices or "things" in 2025 that generate 79.4 ZB of data [3]. However, in the face of massive data, traditional information systems need more processing units and storage devices to maintain and manage them, which requires a higher cost. Fortunately, cloud storage has played a revolutionary role in data storage [4]. With this storage model, users do not maintain servers, storage, and network devices locally, but simply outsource storage management to cloud server providers (CSPs) [5], who are typically responsible for maintaining physical security and availability of data, such as using encryption schemes to ensure that data is confidential and not leaked, while reducing storage costs and achieving greater economic benefits for data owners [6]. However, when data is outsourced to cloud storage [7], there are still serious security problems, which may not only leak users' privacy [8], but also lead to users' property loss and threaten their life security. Overall, the current cloud storage solution has the following security issues.

1) Whether the cloud server maintains data integrity: users only know that the data is stored in the "cloud", but do not know where and how it is stored. This "secret" will naturally lead to decreased acceptance of safety. While the CSP promises good privacy, being subject to some force majeure, such as earthquakes, floods, and power outages, can lead to data loss.

2) CSP is not entirely trustworthy: to save on storage resources, CSP can remove data rarely retrieved without notifying the data owner (user). Data can be manipulated, but CSP is not aware of the risks of violating users' privacy, and may lead to the loss of users' property, and even threaten the personal safety of users. For example, a cloud storage provider called LinkUp (MediaMaSi) lost 45% of its storage customer data due to a system administrator error, eventually causing [9] to shut down.

In view of the above threats, this paper builds a blockchain-based multi-cloud data storage solution and aggregated digital signatures to solve security problems:

Blockchain in Multiple CSPs: Create a blockchain replica of user data for each CSP that can be tamper-proof.

Proxy Aggregated Signature (PAS) Scheme: This scheme can compress blockchain storage space and reduce communication bandwidth.

The rest of the paper is organized as follows. Based on literature research, Section 2 introduces basic knowledge of cloud storage and blockchain. Section 3 introduces the system model. Section 4 introduces the definition of proxy signatures. Section 5 describes the signature scheme of proxy aggregation. Section 6 analyses the plan. Section 7 evaluates the signature length and time required of PAS scheme.

2. **Related Work.**

The Byzantine problem is a significant issue in distributed systems. It involves achieving reliable information exchange and decision-making in an untrusted environment. Research on the Byzantine problem holds essential significance in constructing secure and dependable distributed systems, as well as in fields such as blockchain technology, where it can ensure data consistency and security in environments with partial trust. Driscoll *et al.* [10] discussed typical failure properties from a practitioner's perspective and proposed solutions addressing the Byzantine problem.

Efanov and Roschin [11] believe that blockchain is a distributed database containing transaction records that are shared among participating members. Each transaction requires consensus confirmation from the majority of members, preventing fraudulent transactions from being collectively confirmed. Once a record is created and accepted by the blockchain, it cannot be changed or disappear, ensuring the security and trustworthiness of the data.

Rashid and Chaturvedi [12] introduced how cloud computing can provide reliable and cost-effective services in various applications, exploring various cloud computing services, applications, and characteristics. They also discussed cloud computing service models and their advantages. Cloud storage is essentially a cloud computing system [13] where users can store and share data on the Internet, and can be used for simple data storage such as enterprise databases and local hard drives. In cloud storage, data is stored in third parties, not dedicated servers used in traditional network data storage. A user is looking at a virtual server when storing data, but such a place does not exist in reality. Its advantages are unlimited storage space, secure, simple, efficient file access and low cost of use. Cloud storage is based on a virtualized infrastructure and consists of four layers, as shown in Figure 1.

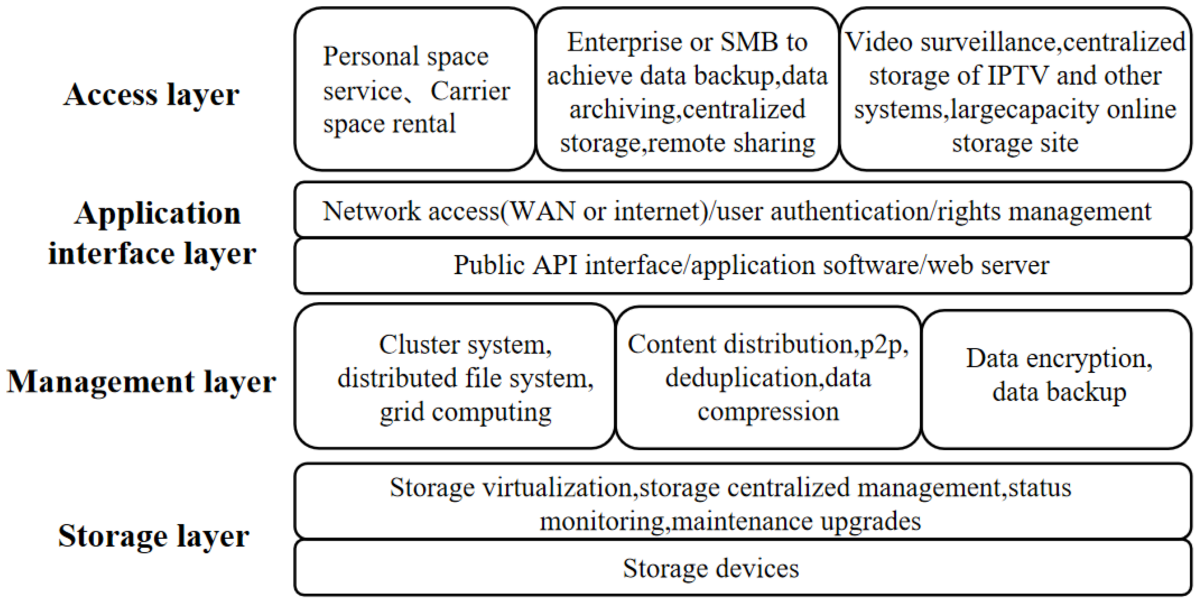

Access layer
Personal space service、Carrier space rental
Enterprise or SMB to achieve data backup,data archiving,centralized storage,remote sharing
Video surveillance,centralized storage of IPTV and other systems,largecapacity online storage site
Application interface layer
Network access(WAN or internet)/user authentication/rights management
Public API interface/application software/web server
Management layer
Cluster system, distributed file system, grid computing
Content distribution,p2p, deduplication,data compression
Data encryption, data backup
Storage layer
Storage virtualization,storage centralized management,status monitoring,maintenance upgrades
Storage devices

FIGURE 1. Structure of cloud storage

1) The storage layer is the fundamental part of cloud storage that consists of a large number of storage devices and is managed and maintained by a unified storage device management system.

2) The management layer is the core part of cloud storage used to realize collaboration between storage devices.

3) The application interface layer is the most flexible part of cloud storage, and various interfaces can provide a range of services.

4) The last is the level of access through which users can access and enjoy cloud services. As stated above, cloud storage refers to services that integrate devices and servers, not devices. Cloud storage services are available, but there is no security mechanism for data transfer and storage of cloud storage. Shahid *et al.* [14] specifically focus on data security issues in a multi-cloud environment and propose a distributed storage solution. To counter related key attacks, pollution attacks, known ciphertext attacks, and known plaintext attacks, the approach divides data into two categories: regular data and sensitive data, with sensitive data further divided into two parts. Each part undergoes encryption and distribution across multiple clouds, while regular data is uploaded in encrypted form

to a single cloud. During the decryption stage, sensitive data is merged from multiple clouds, which involves relatively intricate operations. Alouffi *et al.* [15] conducted a review of cloud computing security-related research published between 2010 and 2020 in reputable digital libraries. They expounded upon seven major security threats that cloud computing services face. Their research findings indicated that data tampering and leakage were among the extensively discussed topics. The paper identified blockchain as a collaborative technology to mitigate security concerns.

Due to the interconnectivity of distributed networks [16] and the importance of cloud, Namasudra and Deka [17] believe that blockchain technology is the best way to provide security for cloud storage. Wu *et al.* [18] propose an enhanced protocol for identity verification in a distributed cloud computing architecture. Blockchain technology [19] can be the best security solution in the cloud environment because it allows quick communication and significantly reduces the required processing resources. In other words, it is not possible to modify or delete the transaction information by using the security inherent in blockchain technology. Ismail *et al.* [20] believe that distributed data book can support anomaly invariance and data security even if shared across all nodes in the cloud. Because the encryption chain algorithm is adopted in the blockchain block, the privacy of the data is more liable to be protected. Because of these characteristics, blockchains are the most likely candidates for providing cloud data security. Zhang *et al.* [21] present a blockchain-based multi-cloud storage data auditing scheme to protect data integrity and accurately arbitrate service disputes. Xiong *et al.* [22] propose an an ECDSA based verification scheme that can achieve efficient large-scale batch verificationTo ensure lower storage, computation and communication costs, we have designed the following scheme.

3. System Model.

3.1. Blockchain-Based Multiple CSPs.

Our storage model is shown in Figure 2. Unlike the traditional cloud storage model, we build a multi-cloud storage provider environment to replace a single cloud storage provider.

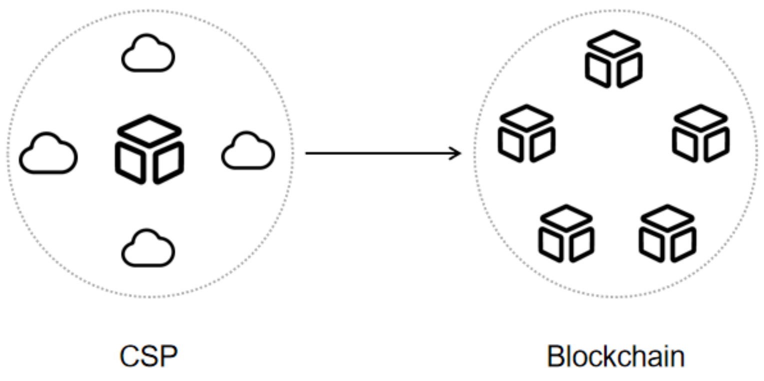


Figure 2. Storage model

The data blockchain is built on the basis of several cloud service providers. Each provider, in turn, consists of multiple cloud storage servers that can store unlimited amounts of data. In addition, each provider has a copy of the blockchain. The data is stored in the blockchain and the storage confirmation is executed by the storage transaction. Each client sends data to the cloud server. After receiving the data, the cloud server writes the signed data to the blockchain according to the PAS scheme.

3.2. Blockchain Storage.

3.2.1. *Advantages of Blockchain Storage.* First, blockchain storage stores data [23] on tens of millions of nodes worldwide with high reliability, rather than using multi-copy mode, but using more advanced redundant encoding modes [24]. This effectively avoids the negative effects of single point of failure. Distributed data storage as the main direction of future storage technology development can improve system reliability, availability and access efficiency and is easy to expand [25].

Second, the costs are relatively low. The basic reason for the low cost of blockchain storage is that blockchain technology effectively solves the problem of removing data duplication. In addition, the construction cost of each storage node is also very low. The edge node architecture used in the blockchain has lower hardware requirements and much lower costs than setting up a central data storage center [26].

Third, the storage form and consensus mechanism of the blockchain ensure that the manipulated data of a single or even multiple nodes does not affect the databases of other nodes and the manipulated data can be synchronized with the correct data. As long as an attacker does not control more than 50% nodes or has 50% computing power, he cannot really manipulate the blockchain [27]. Attackers are unlikely to crack this system.

3.2.2. *Disadvantages of Blockchain Storage.* High Storage Cost: Due to the need for each node on the blockchain to store all blockchain data, and to store historical data, information cannot be deleted, the storage requirement of each node on the blockchain will continue to increase, resulting in increased storage device costs and high storage resource consumption [28].

Slow Processing Speed: Blockchain transactions require all nodes in the chain to participate, and nodes also need to be encrypted and transmitted over the network, which can make the transaction speed quite slow. Slow processing results in low parallelism, which will limit the scale and user experience of the blockchain.

3.2.3. *Our Solution.* This article proposes a PAS system. This schema can combine the signatures of all users into one signature, improving the efficiency of signature verification. In addition, it can compress storage space and reduce communication bandwidth. The length and transmission efficiency of signatures have always been a concern. This article uses a short signature scheme that can effectively reduce network traffic and avoid network congestion. The user's output signature can be integrated into the digital signature via a central gateway or cloud server and stored in the blockchain in a multi-cloud server environment, as shown in Figure 3. The administrator of each data block generates proxy signatures with corresponding keys for each data block, and then aggregates all proxy signatures to obtain the final signed proxy aggregation signature. The cloud server performs proxy signatures and sends the final signature to the blockchain. Administrators only need to use their own private key authentication to view data.

4. **Preparatory Knowledge.**

4.1. **Definition of Proxy Signature.** The proxy signature is defined as follows:

1) Creation of system parameters: Key Generation Center (KGC), input security parameter k, and output system public parameter params.

2) Key pair generation: KGC input user ID and system public parameter params, output user private key S_{ID} and public key Q_{ID}.

3) Proxy authorization: Input the system public parameters params, the key pair of the original signer and the authorization file w, and output the proxy authorization certificate W. Among them: w contains the identity information of the original signer and the proxy

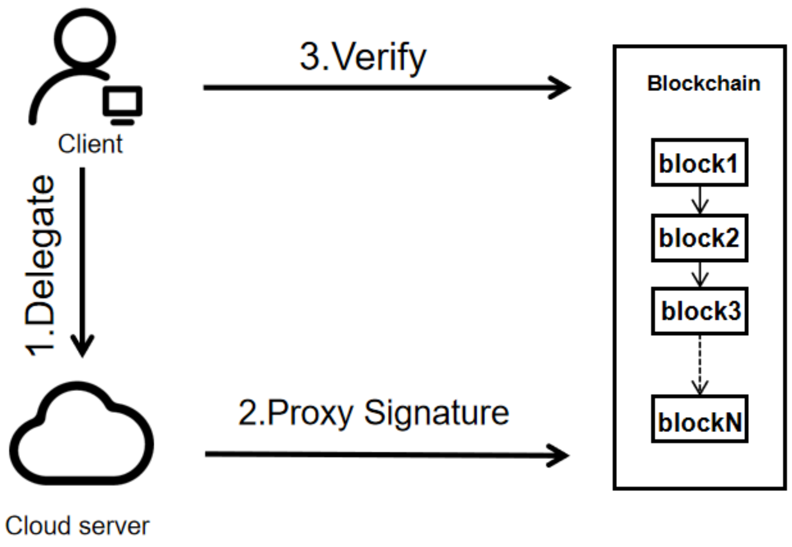


FIGURE 3. Signature scheme

signer, the description of the authorization relationship, the scope and duration of the messages allowed to be signed, etc.

4) Proxy authorization verification: Input system public parameters params, original signer U_0 identity ID_0 and proxy authorization certificate W. The verifier verifies whether W is a valid proxy authorization certificate.

5) Proxy key generation: Proxy signer U_i enters system public parameters params, proxy authorization certificate W, and outputs X as the proxy signature private key.

6) Proxy signature: Input system public parameters params, proxy authorization document w, proxy signature private key X and message m to be signed, output signature message σ.

7) Proxy signature verification: Enter the system public identity parameter params, original signer U_0 identity ID_0, proxy signer U_i identity ID_i, authorization certificate W and signature information σ, return "1" means verification passed, return "0" means verification failed.

4.2. **Security Assumptions.** The hard problems of this article are as follows:

Definition 4.1. *the $k-CAA$ problem. G_1 is a cyclic additive group of order a safe large prime q. For an integer k and $x \in Z_q^*$, given $\{e_1, e_2, \dots, e_k\} \in Z_q^*$. P, $xP \in G_1$ and $\{P/(x+e_1), P/(x+e_2), \dots, P/(x+e_k)\}$, compute $(e, P/(x+e))$, where $e \in Z_q^*$, $e \notin \{e_1, e_2, \dots, e_k\}$.*

Definition 4.2. *Elliptic curve $Diffie-Hellman$ problem (CDH). Given an elliptic curve group G, generate element P and two points a, $b \in Z_q^*$. Compute abP using the known aP and bP.*

Definition 4.3. *Bilinear pairing: in the system, G_1 and G_2 are groups of order q with generators P and Q, respectively. G_r is also a group of order q. The constructed signature uses the bilinear mapping e: $G_1 \times G_2 \to G_r$, with the following properties:*

*-**Bilinear**: $\forall a, b \in Z_q^*$, $e(aP, bQ) = e(P, Q)ab = e(abP, Q) = e(P, abQ)$.*

*-**Non-degenerate**: $e(P, Q) \neq 1$.*

*-**Computability**: $e(P, Q)$ is computable.*

e: If $G_1 \neq G_2$, $G_1 \times G_2 \to G_r$ is an asymmetric bilinear pair; otherwise symmetric.

4.3. **Adversary Model.** On behalf of the original signer, the proxy signer can create a valid proxy signature for the designated verifier, but a proxy signature cannot be generated by the original signer and other third parties, except for the designated verifier. This situation occurs because the outsider does not know either the proxy private key or the private key of the designated verifier. Before discussing the strong unforgeability of our proxy signature scheme in the random oracle model, we first addressed the following types of adversaries that could exist in the system:

1) Type 1: Attacker A_1 knows both the public key of the original signer and the public key of the proxy signer.

2) Type 2: Attacker A_2 knows not only the public key of the original signer and the public key of the proxy signer, but also the private key of the original signer, but not the private key of the proxy signer.

3) Type 3: Attacker A_3 not only knows the public key of the original signer and the public key of the proxy signer, but also knows the private key of the proxy signer, but not the private key of the original signer.

From the description of the capabilities possessed by the above three types of attackers, it is clear that when the proxy signature scheme is existentially unforgeable for attacker A_2 and attacker A_3, then it is also unforgeable for attacker A_1. Therefore, in this paper, only attacker A_2 of type 2 and attacker A_3 of type 3 are considered in the subsequent security models and security proofs.

5. **Signature Scheme Construction.**

5.1. **Basic Construction.** The proxy aggregation signature scheme is as follows:

1) System Parameters Generation.

When the security parameter λ is given, the system key generation centerKGC first selects a secure bilinear mapping e: $G_1 \times G_1 \to G_2$, and G_1 is an additive group with order secure prime q and generating element p, and G_2 is a multiplicative group with the same order secure prime q. Then it selects two collision-resistant secure hash functions H_1: $\{0, 1\}^* \to G_1$, H_2: $\{0, 1\}^* \to Z_q^*$. Finally, expose the system parameters params = $\{q, G_1, G_2, e, p, H_1, H_2\}$. Select the system master key s.

2) Key Pair Generation.

Taking the system public parameter params as input, KGC then randomly picks $S_0 \in Z_q^*$ and computes $Q_0 = S_0 p$ and outputs (Q_0, S_0) as the key pair of its original signer U_0. KGC randomly selects $S_i \in Z_q^*$, compute $Q_i = S_i p$ and output (Q_i, S_i) as the key pair of the proxy signer U_i.

3) Delegate Authorization.

The original signer U_0 takes as input the system disclosure parameter params and first randomly selects $r \in Z_q^*$, compute $R = rp$, and R is used as the identifier of the proxy signer U_i, which can be used to revoke the authorization and broadcast the public. Then calculate $h_0 = H_1(ID_0, w, R)$, where w is the authorization file containing an explicit description of the authorization relationship, the identity information of the original signer U_0 and the proxy signer U_i, the scope and duration of the messages allowed to be signed, etc. Finally, $d = (S_0 + r)h_0$ is calculated and (R, d) is output as the authorization permission of the original signer to the proxy signer.

4) Proxy Authorization Verification.

The proxy signer U_i receives the authorization permission and enters the system public parameter params, verifies that $e(d, p) = e(h_0, Q_0 + R)$ holds, and accepts the authorization permission if the equation holds.

5) Proxy Key Generation.

The proxy signer U_i, according to the system public parameter params, calculates $X_i = d + S_i h_0$, and outputs X_i as the proxy signature key.

6) Proxy Signature.

The proxy signer U_i does the following for a given message $m_i \in \{0, 1\}^*$: (1) Compute $h_i = H_2(ID_0, ID_i, m_i, R)$; (2) Calculate $\sigma_i = X_i/(h_i + S_i)$; Then σ_i is the signature of the proxy signer U_i on the message m_i.

7) Signature Verification.

For a given message/signature pair (m_i, σ_i), verification is performed by the following steps:

(1) Compute $h_0 = H_1(ID_0, w, R)$;

(2) Compute $h_i = H_2(ID_0, IDi, mi, R)$;

(3) Verify whether the equation $e(\sigma_i, h_i p + Q_i) = e(h_0, Q_0 + Q_i + R)$ holds, if the equation holds, the signature verification passes; if the equation does not hold, the signature verification fails.

5.2. **Aggregation.** The first five algorithms are the same as in Section 5.1.

Aggregate Signature

The proxy signer performs the following operations for a given message $m_i \in \{0, 1\}^*$, $(1 \le i \le n)$:

(1) Compute $h_i = H_2(ID_0, ID_i, m_i, R)$;

(2) Calculate $\sigma_i = X_i/(h_i + S_i)$, $\sigma_{agg} = \sum_i^n \sigma_i$; and then σ_{agg} is the proxy aggregation signature.

Aggregate Verification

For message/signature pairs (m_i, σ_{agg}), $(1 \le i \le n)$, verification is performed by the following steps:

(1) Compute $h_0 = H_1(ID_0, w, R)$;

(2) Compute $h_i = H_2(ID_0, ID_i, m_i, R)$;

(3) Verify that the equation $e(\sigma_{agg}, \sum_i^n (h_i p + Q_i)) = e(nh_0, \sum_i^n (Q_0 + Q_i + R))$ holds, if the equation holds, then the signature verification passes; if the equation does not hold, then the signature verification fails.

The detailed procedure of the above stages is shown in Figure 4.

6. **Scheme Analysis.**

6.1. **Correctness Analysis.** The correctness of the authorization license is analyzed as follows:

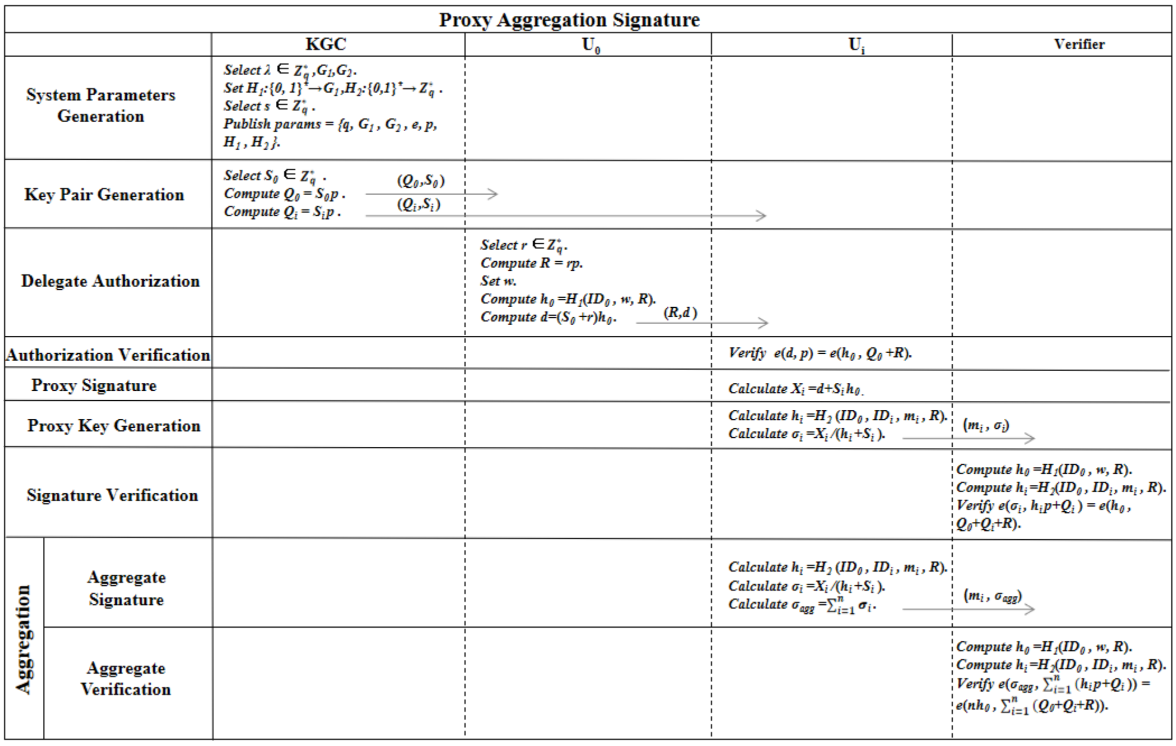
$$e(d, p) = e((S_0 + r)h_0, p) = e(h_0, (S_0 + r)p) = e(h_0, Q_0 + R)$$

The analysis of the correctness of the single signature verification equation is as follows:

$$\begin{aligned} e(\sigma_i, h_i p + Q_i) &= e(X_i/(h_i + S_i), p(h_i + S_i)) = e(X_i, p) \\ &= e(d + S_i h_0, p) = e((S_0 + r)h_0 + S_i h_0, p) \\ &= e((S_0 + r + S_i)h_0, p)e(h_0, Q_0 + Qi + R) \end{aligned}$$

The analysis of the correctness of the aggregated signature verification equation is as follows:

$$\begin{aligned} e(\sigma_{agg}, \textstyle\sum_i^n (h_i p + Q_i)) &= e(\textstyle\sum_i^n X_i/(h_i + S_i), p(h_i + S_i)) = e(\textstyle\sum_i^n X_i, np) \\ &= e(\textstyle\sum_i^n (d + S_i h_0), np) = e(\textstyle\sum_i^n ((S_0 + r)h_0 + S_i h_0), np) \\ &= e(\textstyle\sum_i^n ((S_0 + r + S_i)h_0), np)e(nh_0, \textstyle\sum_i^n (Q_0 + Qi + R)) \end{aligned}$$



Proxy Aggregation Signature				
	KGC	U_0	U_i	Verifier
System Parameters Generation	Select $\lambda \in Z_q^*, G_1, G_2$. Set $H_1:\{0,1\}^* \to G_1, H_2:\{0,1\}^* \to Z_q^*$. Select $s \in Z_q^*$. Publish params = $\{q, G_1, G_2, e, p, H_1, H_2\}$.			
Key Pair Generation	Select $S_0 \in Z_q^*$. Compute $Q_0 = S_0 p$. Compute $Q_i = S_i p$. (Q_0, S_0) (Q_i, S_i)			
Delegate Authorization		Select $r \in Z_q^*$. Compute $R = rp$. Set w. Compute $h_0 = H_1(ID_0, w, R)$. Compute $d=(S_0+r)h_0$. (R,d)		
Authorization Verification			Verify $e(d, p) = e(h_0, Q_0+R)$.	
Proxy Signature			Calculate $X_i = d+S_i h_0$.	
Proxy Key Generation			Calculate $h_i = H_2(ID_0, ID_i, m_i, R)$. Calculate $\sigma_i = X_i/(h_i+S_i)$.	(m_i, σ_i)
Signature Verification				Compute $h_0 = H_1(ID_0, w, R)$. Compute $h_i = H_2(ID_0, ID_i, m_i, R)$. Verify $e(\sigma_i, h_i p+Q_i) = e(h_0, Q_0+Q_i+R)$.
Aggregation: Aggregate Signature			Calculate $h_i = H_2(ID_0, ID_i, m_i, R)$. Calculate $\sigma_i = X_i/(h_i+S_i)$. Calculate $\sigma_{agg} = \sum_{i=1}^{n} \sigma_i$.	(m_i, σ_{agg})
Aggregation: Aggregate Verification				Compute $h_0 = H_1(ID_0, w, R)$. Compute $h_i = H_2(ID_0, ID_i, m_i, R)$. Verify $e(\sigma_{agg}, \sum_{i=1}^{n}(h_i p+Q_i)) = e(nh_0, \sum_{i=1}^{n}(Q_0+Q_i+R))$.

FIGURE 4. Proxy aggregation signature

6.2. **Security Analysis. Theorem 1** For the second class of attackers, the scheme in this paper is existentially unforgeable under the stochastic prediction machine model and the k-CAA hard problem assumption. Assume that the adversary A_2 after a finite number of interrogations breaks the paper's scheme in polynomial time t by a non-negligible advantage ϵ, noting q_{H_i} and t_{H_i} are respectively the adversary A_2 interrogation $H_i (i = 1, 2)$ the number of times and the time required for one interrogation of the prediction machine, and denote q_R and t_R are the number of authorization identifier interrogations and the time required for one interrogation, respectively, denoted as q_s and t_s are the number of proxy signature queries and the time required for one query, respectively, then there exists an algorithm C with probabilistic polynomial time $t' < t + (q_R t_R + q_s t_s + 2q_{H_1} t_{H_1} + 2q_{H_2} t_{H_2})$within a non-negligible advantage

$$\epsilon' \geq (1 - \frac{1}{2^\lambda})\frac{q_{H_2}-k}{q_{H_2}} - (\frac{k}{q_{H_2}})^{q_s}$$

Solving the k-CAA hard problem.

Proof. Assume that the instance of the k-CAA hard problem challenged by C is given by $\{e_1, e_2, \ldots, e_k\} \in Z_q^*$, $\{P/(x+e_1), P/(x+e_2), \ldots, P/(x+e_k)\}$ and $bP \in G_1$, compute $(e, P/(b+e))$, where $e \in Z_q^*$, $e \notin \{e_1, e_2, \ldots, e_k\}$. Let the security parameter be λ. Algorithm C first runs the system parameter building algorithm with the output params $= \{\lambda, q, G_1, G_2, e, P, H_1, H_2\}$. In the game C selects the original signer A identity as ID_A, the proxy signer B identity is ID_B, and randomly selects $S_A \in Z_q^*$, and computes $Q_A = S_A P$ by combining (S_A, Q_A) as the key pair of its original signer A and set the public key of the proxy signer B, $Q_B = bP$. Send (S_A, Q_A), Q_B and params to A_2. C uniformly and randomly selects $q_{H_2} - k$ numbers $f_1, f_2, \ldots, f_{q_{H_2}-k} \in Z_q^*$, define the set $E = \{e_1, e_2, \ldots, e_k\}$, the set $F = \{f_1, f_2, \ldots, f_{q_{H_2}-k}\}$, $E \cap F = \emptyset$, $\Omega = E \cup F$. For the sake of narrative simplicity, assume that A_2 will not make the same queries to the prediction machine, and that signature queries and forged signatures have been made at the time of H_1 and H_2 interrogations, and all record lists are initialized to empty. C allows A_2 to make the following adaptive queries.

1) Authorization Identifier Query.

C maintains a record structure as an array $(ID_A,\, ID_B,\, w,\, r,\, R)$ list L_R. When submits to C a list of records for which the original signer identity is ID_A and the identity of the proxy signer as ID_B. C checks the list L_R whether the corresponding value of the query already exists in the list $(ID_A,\, ID_B,\, w,\, r,\, R)$, and returns the corresponding value $(w,\, R)$ to A_2; otherwise C generates the authorization message w, randomly selects $r \in Z_q^*$, compute $R = rP - Q_B$, C returns $(w,\, R)$ to A_2, while returning $(ID_A,\, ID_B,\, w,\, r,\, R)$ is recorded into the list L_R.

2) H_1 Query.

C maintains a record structure as an array $(ID_A,\, ID_B,\, w,\, R,\, v,\, h_1)$ of list L_{H_1}. When A_2 submits a list to C of $(ID_A,\, ID_B,\, w,\, R)$ of H_1 query, C checks the list L_{H_1} whether the corresponding value of the query already exists in $(ID_A,\, ID_B,\, w,\, R,\, v,\, h_1)$, and returns the corresponding value if it exists h_1 to A_2 ; otherwise C uniformly and randomly selects $v \in Z_q^*$, compute $h_1 = vP$, which will h_1 return to A_2 and also returns $(ID_A,\, ID_B,\, w,\, R,\, v,\, h_1 = vP)$ is recorded into the list L_{H_1}.

3) H_2 Query.

C maintains a record structure as an array $(ID_A,\, ID_B,\, m,\, R,\, h_2)$ of list L_{H_2}. When A_2 submits a list of $(ID_A,\, ID_B,\, m,\, R)$ of H_2 query, C checks the list L_{H_2} whether the corresponding value of the query already exists $(ID_A,\, ID_B,\, m,\, R,\, h_2)$ and if it exists, returns the corresponding h_2 to A_2; otherwise a uniformly random selection of $h_2 \in \Omega$ returns the value given by A_2, also return $(ID_A,\, ID_B,\, m,\, R,\, h_2)$ is recorded to the list L_{H_2}. Clearly here, the probability of $h_2 \in E$ is $\frac{k}{q_{H_2}}$, and $h_2 \in F$ has probability $\frac{q_{H_2}-k}{q_{H_2}}$.

4) Proxy Signature Query.

When A_2 submits a report on $(m,\, ID_A,\, ID_B,\, R)$ of a proxy signature query, then C gets the record from the list L_{H_1} of records $(ID_A,\, ID_B,\, w,\, R,\, v,\, h_1)$, and C gets the corresponding record $(ID_A,\, ID_B,\, w,\, R,\, v,\, h_2)$ from the list L_{H_2}, C gets the corresponding record from the list L_R to get the record $(ID_A,\, ID_B,\, w,\, r,\, R)$, and then:

① If $h_2 \in E$, that is, there exists $e_j = h_2 \in E$, compute $\sigma = v(S_A + r)P/(b + e_j)$, returning σ to A_2.

② If $h_2 \in F$, "$\perp$" is returned and this event is noted as $Event1$, "$\perp$" means empty. It is worth noting that if $h_2 \in E$, the constructed proxy signature is able to pass the verification equation, and its soundness is guaranteed by the following equation.

$$e(\sigma,\, h_2P + Q_B) = e(v(S_A + r)P/(b + e_j),\, e_jP + Q_B) = e(vP,\, Q_A + Q_B + R).$$

$R = rP - Q_B$ After polynomially bounded subadaptive interrogation, finally A_2 stop interrogation and output a valid message proxy signature pair $(m^*,\, \sigma^*)$. C From the list L_{H_2} obtains the records $(ID_A,\, ID_B,\, m^*,\, R^*,\, h_2^*)$, and if $h_2 \in E$, then the forged signature is invalid and the event is noted as $Event2$; otherwise if $h_2 \in F$, get the record from the list L_R record from the list $(ID_A,\, ID_B,\, w,\, r^*,\, R^*)$, and from the list L_{H_1} get the record from the list $(ID_A,\, ID_B,\, w,\, R^*,\, v^*,\, h_1^* = v^*P)$. The following verification equation is available.

$$\begin{gathered} e(\sigma^*,\, h_2^*P + Q_B) = e(h_1^*,\, Q_A + Q_B + R^*) \\ e(\sigma^*,\, h_2^*P + bP) = e(v^*P,\, (S_A + r^*)P) \\ e((h_2^* + b)\sigma^*,\, P) = e((S_A + r^*)v^*P,\, P) \\ ((h_2^* + b)\sigma^* = (S_A + r^*)v^*P \\ P/(b + h_2^*) = \sigma^*/((S_A + r^*)v^*) \end{gathered}$$

That is, the output $\sigma^*/((S_A + r^*)v^*)$ as the answer to the difficult problem, and thus C solves the $k - CAA$ problem instance.

The following is an analysis of the time and advantages required for C to successfully solve difficult problems.

1) For H_1 and H_2 the answers to the prophecy machine queries are uniformly and independently distributed between G_1 and Z_q^*.

2) An instance where C solves the $k - CAA$ hard problem only if both $Event1$ and $Event2$ do not occur. The probability that $Event1$ does not occur all the time is $(\frac{k}{q_{H_2}})^{q_s}$, the probability that $Event2$ does not occur is $\frac{q_{H_2}-k}{q_{H_2}}$ then the probability that both $Event1$ and $Event2$ do not occur is obtained as:

$$Pr(Event1 \cap Event2) \geq \frac{q_{H_2}-k}{q_{H_2}}(\frac{k}{q_{H_2}})^{q_s}$$

When A_2 no forgery is made H_2 query and forging a valid signature its probability of occurrence is $1/2^\lambda$, so C's advantage in the game is estimated to be satisfied.

$$\epsilon^{'} \geq (1 - \frac{1}{2^\lambda})\frac{q_{H_2}-k}{q_{H_2}} - (\frac{k}{q_{H_2}})^{q_s}$$

Running time is met.

$$t^{'} < t + (q_R t_R + q_s t_s + 2q_{H_1} t_{H_1} + 2q_{H_2} t_{H_2})$$

Thus C successfully solves an instance of the $k - CAA$ problem in polynomial time t' with a non-negligible advantage $\epsilon^{'}$, which contradicts the difficulty of the $k - CAA$ problem. So the scheme in this paper is existentially unfalsifiable against the second class of adversaries.

Theorem 2 For the third class of attackers, the scheme in this paper is existentially unforgeable under the assumptions of the stochastic prediction machine model and the CDH hard problem. Assumes that the adversary A_3 after a finite number of queries breaks the scheme of this paper in polynomial time t with a non-negligible advantage ϵ, noting q_{H_i} and t_{H_i} are the adversary A_3 queries, respectively $H_i (i = 1, 2)$ the number of times and the time required for one interrogation of the prophecy machine, and denote q_R and t_R are the number of authorized identifier queries and the time required for one query, respectively, and denote q_s and t_s are the number of proxy signature queries and the time required for one query, respectively, and $\delta \in (0, 1/2)$, then there exists an algorithm C with probabilistic polynomial time $t^{'} < t + (q_R t_R + q_s t_s + 2q_{H_1} t_{H_1} + 2q_{H_2} t_{H_2})$ within a non-negligible advantage $\epsilon^{'} \geq \delta(1 - \delta)^{q_s}(1 - 1/2^\lambda)$ to solve the CDH hard problem.

Proof. Assume that the instance of the CDH hard problem for the C challenge is: Given $ap \in G_1$ and $bp \in G_1$, compute abp, where $a, b \in Z_q^*$. Let the security parameter be λ. Algorithm C first runs the system parameter building algorithm with the output params $= \{\lambda, q, G_1, G_2, e, p, H_1, H_2\}$. In the game C selects the original signer A identity as ID_A, the proxy signer B identity is ID_B, randomly selects $S_B \in Z_q^*$, and computes $Q_B = S_B p$, which will be (Q_B, S_B) as the key pair of its proxy signer B and set the public key $Q_A = ap$ of the original signer A. Send (Q_B, S_B), Q_A and params to A_3. For the sake of narrative simplicity, assume that A_3 will not make the same queries to the propagator and that H_1 and H_2 queries have already been made when making signature queries and forging signatures, and that all record lists are initialized to empty. C allows A_3 to make the following adaptive queries.

1) Authorization Identifier Query.

C maintains a record structure as an array (ID_A, ID_B, w, c, r, R) list L_R. When A_3 submits an authorization parameter query to C for an original signer identity of ID_A and a proxy signer identity of ID_B, C checks the list L_R whether the corresponding value of the query already exists in the list (ID_A, ID_B, w, c, r, R), the corresponding values $(w,$

R) are returned to A_3 if they exist; otherwise C generates the authorization message w and randomly selects $r \in Z_q^*$, set $c = 1$ with δ probability and compute $R = rp$, set $c = 0$ with $1 - \delta$ probability and compute $R = rp - ap$. Finally C returns (w, R) to A_3, and also returns (ID_A, ID_B, w, c, r, R) recorded into the list L_R in the list.

2) H_1 Query.

C maintains a record structure as an array $(ID_A, ID_B, w, R, v, h_1)$ of lists of L_{H_1}. When A_3 submits a record to C about (ID_A, ID_B, w, R) of H_1 query, C checks the list L_{H_1} whether the corresponding value of the query $(ID_A, ID_B, w, R, v, h_1)$, and returns the corresponding value h_1 to A_3 if it exists; otherwise, C randomly selects $v \in Z_q^*$, compute $h_1 = vbp$ and returns h_1 to A_3 while returning $(ID_A, ID_B, w, R, v, h_1 = vbp)$ is recorded into the list L_{H_1}.

3) H_2 Query.

C maintains a record structure as an array (ID_A, ID_B, m, R, h_2) of list L_{H_2}. When A_3 submits a record about (ID_A, ID_B, m, R) of the H_2 query, C checks the list L_{H_2} whether there already exists a query corresponding to the value (ID_A, ID_B, m, R, h_2), and if it exists, returns the corresponding h_2 to A_3; otherwise, uniformly selects randomly $h_2 \in Z_q^*$ is returned to A_3, while (ID_A, ID_B, m, R, h_2) is recorded to the list L_{H_2}.

4) Proxy Signature Query.

When A_3 is submitted on (m, ID_A, ID_B, R, X) for a proxy signature query, then C gets the corresponding record h_2 from the list L_{H_2}, C gets the corresponding record (ID_A, ID_B, w, c, r, R) from the list L_R. If $c = 0$ then calculate $\sigma = (S_B + r)vbp/(S_B + h_2)$, return σ to A_3; otherwise, return "$\perp$" and note this event as $Event1$. The constructed proxy signature is able to pass the verification equation and its soundness is guaranteed by the following equation.

$$e(\sigma, h_2p + Q_B) = e((S_Bvbp + rvbp)/(S_B + h_2), h_2p + Q_B) = e(vbp, Q_A + Q_B + R)$$
$$R = rp - Q_A$$

After polynomially bounded subadaptive interrogation, finally A_3 stops interrogating and outputs a valid message proxy signature pair (m^*, σ^*). From the list L_R, get the record $(ID_A, ID_B, w, c^*, r^*, R^*)$, if $c = 0$, the forgery is invalid and the event is noted as $Event2$.

If $c = 1$, C gets the record $((ID_A, ID_B, w, R^*, v^*, h_1^* = v^*bp)$from the list L_{H_1}, get the record $(ID_A, ID_B, w, m^*, R^*, h_2^*)$ from the list L_{H_2}, with the following verification equation.

$$e(\sigma^*, h_2^*p + Q_B) = e(h_1^*, Q_A + Q_B + R^*)$$
$$e(\sigma^*, h_2^*p + S_Bp) = e(v^*bp, (S_B + r^* + a)p)$$
$$e(\sigma^*, (h_2^*p + S_B)p) = e(v^*bp, (S_B + r^* + a)p)$$
$$e((h_2^*p + S_B)\sigma^*, p) = e((S_B + r^* + a)v^*bp, p)$$

So C can be calculated successfully.

$$abp = (h_2^*p + S_B)\sigma^*/v^* - S_Bbp - r^*bp$$

Output:

$$(h_2^*p + S_B)\sigma^*/v^* - S_Bbp - r^*bp$$

As an example answer to the CDH problem, thus C solves the CDH problem. The following is an analysis of the time and advantages required for C to successfully solve the difficult problem.

1) For H_1 and H_2 the answers to the prophecy machine queries are uniformly and independently distributed between G_1 and Z_q^* respectively.

2) C solves an instance of the CDH problem only if neither $Event1$ nor $Event2$ occurs all the time during the simulation phase. The probability that $Event1$ does not occur all

the time is $(1-\delta)^{q_s}$ and the probability that $Event2$ does not occur is δ. The probability that both $Event1$ and $Event2$ do not occur can be obtained as:

$$Pr(Event1 \cap Event2) \geq \delta(1-\delta)^{q_s}$$

When A_3 does not ask H_1 and a valid signature is forged, this simulation is flawed with a probability of occurrence of $1/2^{\lambda}$, so the advantage of C in the game is satisfied.

$$\epsilon^{'} \geq \delta(1-\delta)^{q_s}(1-1/2^{\lambda})$$

Running time is met.

$$t^{'} < t + (q_R t_R + q_s t_s + 2q_{H_1} t_{H_1} + 2q_{H_2} t_{H_2})$$

Thus C successfully solves an instance of the CDH problem in polynomial time $t^{'}$ with a non-negligible advantage $\epsilon^{'}$, which contradicts the difficulty of the CDH problem. So the scheme in this paper is existentially non-falsifiable for the third class of adversaries.

7. Solution Realization And Efficiency Analysis.

7.1. Program Implementation.

This paper is based on JPBC (Java Pairing-Based Cryptography Library) in an operating system of Windows 10 flagship 64-bit, processor of Intel Core i5-6300HQ, motherboard of ASUS GL552VX, and memory of Kingston DDR4 2400MHz 8GB The solution was implemented in an experimental benchmarking environment with the IntelliJ IDEA development platform.

7.2. Computational Efficiency Analysis.

The computational efficiency and signature length of this schema were compared with several existing proxy signature schemes, these schemes are based on certificateless signature, identity-based signature, the results are presented in Table 1. Where: Sm denotes scalar multiplication operation, E denotes exponential operation, Pr denotes bilinear pair operation, H denotes hash operation, $|G_1|$ denotes the length of the elements on the additive group G_1, $|Z_q^*|$ denotes the length of the elements on Z_q^*.

Table 1. Comparison of computational efficiency and signature lengths

Scheme	Signature Operation	Verify Operation	Signature Length
Reference [29]	2Sm+H+E	4Sm+3Pr+4H	$\|Z_q^*\|$
Reference [30]	2Sm+2Pr+3H	2Sm+6Pr+2E+3H	$\|G_1\|+\|Z_q^*\|$
Reference [31]	2Pr+3E+3H	2Sm+6Pr+2E+3H	$2\|G_1\|+\|Z_q^*\|$
PAS	Sm+H	Sm+2Pr+2H	$\|G_1\|$

As can be seen from Table 1, in the signature generation process, the scheme in this paper performs one Sm operation and one H operation, and its computational efficiency is optimal compared with the scheme in the literature [29, 30, 31]. In the signature verification process, this scheme performs one Sm operation, two H operations and two Pr operations, and its computational efficiency is better than [29, 30, 31]. However, in terms of signature length, the signature length of the scheme in this paper is $|G_1|$, compared to other solutions, it is shorter.

7.3. Operational Efficiency Analysis.

The implementation of this paper's scheme and several existing proxy signature schemes were carried out in the same experimental test benchmark environment, and the operational efficiency was compared after several runs, and the results are shown in Figure 5 and Figure 6.

From Figure 5 and Figure 6, we conducted five comparative experiments on signature time and verification time, respectively. The average total time consumed for the scheme

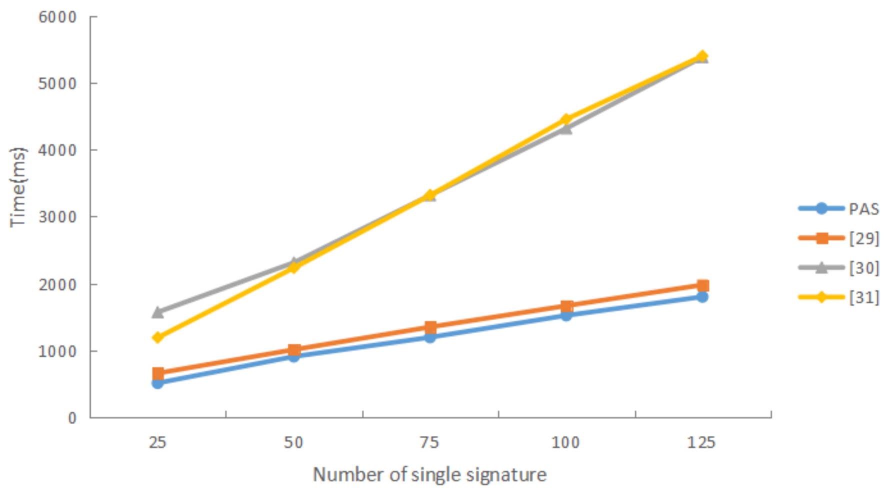


Figure 5. Time cost of signature algorithm

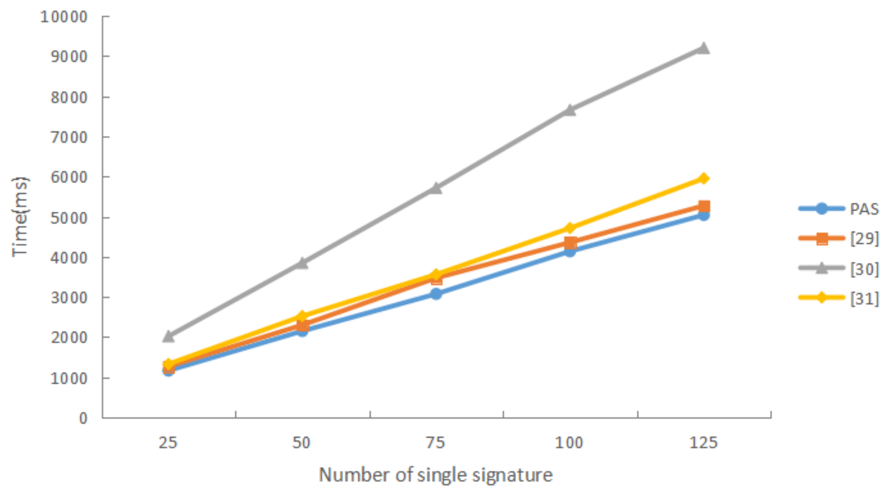


Figure 6. Time cost of verify algorithm

in this paper is reduced by about 13.0% compared with the scheme in [29], about 49.8% compared with the scheme in [30], and about 33.4% compared with the scheme in [31]. From the above analysis, it is clear that the scheme in this paper not only has short signature length and high theoretical calculation efficiency, but also outperforms the scheme in literature [29, 30, 31] in terms of actual operation efficiency.

8. **Conclusions.** In this paper, we propose a multi-cloud storage mechanism based on blockchain in which multi-cloud storage providers build data blockchain. We have developed a PAS solution to ensure that user data is only viewed by executives and compress blockchain storage space. We use blockchain technology to store data in the blockchain. The unchanging nature of blockchain in multi-cloud storage providers ensures data integrity, reliability, and availability.

If we compare the PAS scheme with the public signature scheme, we can see that our scheme performs better in resource saving. Comparing the Certificateless system with traditional ID-based systems, our system consumes less time and has better performance in the energy consumption of communication. We found that our scheme performs better in compressing blockchain storage space compared to other signatures currently in use, saving more resources.

Acknowledgment. This work is partially supported by the natural Science Foundation of Fujian Province of China(2023J01920). The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers, which have improved the presentation.

REFERENCES

[1] S. Madakam and V. Lake, "Internet of things (iot): A literature review," *Journal of Computer and Communications*, vol. 3, no. 05, pp. 164, 2015.

[2] T.-Y. Wu, L. Wang, and C.-M. Chen, "Enhancing the security: A lightweight authentication and key agreement protocol for smart medical services in the ioht," *Mathematics*, vol. 11, no. 17, pp. 3701, 2023.

[3] P. Yang, N. Xiong, and J. Ren, "Data security and privacy protection for cloud storage: A survey," *IEEE Access*, vol. 8, pp. 131723–131740, 2020.

[4] S. Namasudra, "Cloud computing: A new era," *Journal of Fundamental and Applied Sciences*, vol. 10, no. 2, https://doi.org/10.4314/jfas.v10i2.9, 2018.

[5] X. Huang, H. Xiong, J. Chen, and M. Yang, "Efficient revocable storage attribute-based encryption with arithmetic span programs in cloud-assisted internet of things," *IEEE Transactions on Cloud Computing*, pp. 1273–1285, 2021.

[6] I. Odun-Ayo, O. Ajayi, B. Akanle, and R. Ahuja, "An overview of data storage in cloud computing," in *2017 International Conference on Next Generation Computing and Information Systems (ICNGCIS)*. IEEE, 2017, pp. 29–34.

[7] M. Ali, S. U. Khan, and A. V. Vasilakos, "Security in cloud computing: Opportunities and challenges," *Information Sciences*, vol. 305, pp. 357–383, 2015.

[8] R. S. Bhadoria, "Security architecture for cloud computing," in *Cyber Security and Threats: Concepts, Methodologies, Tools, and Applications*. IGI Global, 2018, pp. 729–755.

[9] C. Cachin, I. Keidar, and A. Shraer, "Trusting the cloud," *Special Interest Group on Algorithms & Computation Theory*, vol. 40, no. 2, pp. 81–86, 2009.

[10] K. Driscoll, B. Hall, H. Sivencrona, and P. Zumsteg, "Byzantine fault tolerance, from theory to reality," in *International Conference on Computer Safety, Reliability, and Security*. Springer, 2003, pp. 235–248.

[11] D. Efanov and P. Roschin, "The all-pervasiveness of the blockchain technology," *Procedia Computer Science*, vol. 123, pp. 116–121, 2018.

[12] A. Rashid and A. Chaturvedi, "Cloud computing characteristics and services: a brief review," *International Journal of Computer Sciences and Engineering*, vol. 7, no. 2, pp. 421–426, 2019.

[13] T.-Y. Wu, L. Wang, X. Guo, Y.-C. Chen, and S.-C. Chu, "Sakap: Sgx-based authentication key agreement protocol in iot-enabled cloud computing," *Sustainability*, vol. 14, no. 17, pp. 11054, 2022.

[14] F. Shahid, H. Ashraf, A. Ghani, S. A. K. Ghayyur, S. Shamshirband, and E. Salwana, "Psds–proficient security over distributed storage: a method for data transmission in cloud," *IEEE Access*, vol. 8, pp. 118285–118298, 2020.

[15] B. Alouffi, M. Hasnain, A. Alharbi, W. Alosaimi, H. Alyami, and M. Ayaz, "A systematic literature review on cloud computing security: threats and mitigation strategies," *IEEE Access*, vol. 9, pp. 57792–57807, 2021.

[16] H. Xiong, Z. Kang, J. Chen, J. Tao, C. Yuan, and S. Kumari, "A novel multiserver authentication scheme using proxy resignature with scalability and strong user anonymity," *IEEE Systems Journal*, vol. 15, no. 2, pp. 2156–2167, 2020.

[17] S. Namasudra and G. C. Deka, *Applications of Blockchain in Healthcare*. Springer, 2021.

[18] T.-Y. Wu, F. Kong, Q. Meng, S. Kumari, and C.-M. Chen, "Rotating behind security: an enhanced authentication protocol for iot-enabled devices in distributed cloud computing architecture," *EURASIP Journal on Wireless Communications and Networking*, vol. 2023, no. 1, pp. 36, 2023.

[19] C.-M. Chen, X. Deng, W. Gan, J. Chen, and S. H. Islam, "A secure blockchain-based group key agreement protocol for iot," *The Journal of Supercomputing*, vol. 77, pp. 9046–9068, 2021.

[20] L. Ismail, H. Materwala, and A. Hennebelle, "A scoping review of integrated blockchain-cloud (bcc) architecture for healthcare: applications, challenges and solutions," *Sensors*, vol. 21, no. 11, pp. 3753, 2021.

[21] C. Zhang, Y. Xu, Y. Hu, J. Wu, J. Ren, and Y. Zhang, "A blockchain-based multi-cloud storage data auditing scheme to locate faults," *IEEE Transactions on Cloud Computing*, vol. 10, no. 4, pp. 2252–2263, 2021.

[22] H. Xiong, C. Jin, M. Alazab, K.-H. Yeh, H. Wang, T. R. Gadekallu, W. Wang, and C. Su, "On the design of blockchain-based ecdsa with fault-tolerant batch verification protocol for blockchain-enabled IOMT," *IEEE Journal of Biomedical and Health Informatics*, vol. 26, no. 5, pp. 1977–1986, 2021.

[23] C.-M. Chen, X. Deng, S. Kumar, S. Kumari, and S. H. Islam, "Blockchain-based medical data sharing schedule guaranteeing security of individual entities," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–10, 2021.

[24] W. Liang, Y. Fan, K.-C. Li, D. Zhang, and J.-L. Gaudiot, "Secure data storage and recovery in industrial blockchain network environments," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 10, pp. 6543–6552, 2020.

[25] S. Sekar, A. Solayappan, J. Srimathi, S. Raja, S. Durga, P. Manoharan, M. Hamdi, and G. B. Tunze, "Autonomous transaction model for e-commerce management using blockchain technology," *International Journal of Information Technology and Web Engineering (IJITWE)*, vol. 17, no. 1, pp. 1–14, 2022.

[26] H. Yuan, X. Chen, J. Wang, J. Yuan, H. Yan, and W. Susilo, "Blockchain-based public auditing and secure deduplication with fair arbitration," *Information Sciences*, vol. 541, pp. 409–425, 2020.

[27] M. Saad, J. Spaulding, L. Njilla, C. Kamhoua, S. Shetty, D. Nyang, and D. Mohaisen, "Exploring the attack surface of blockchain: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 1977–2008, 2020.

[28] A. Sosin, O. Ivanova, and S. Vasilyeva, "Prospects for implementing blockchain data storage technology as a process of digital transformation of society," in *2020 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon)*. IEEE, 2020, pp. 1–5.

[29] L. Deng, Z. Hu, Y. Ruan, and T. Wang, "Provably secure certificateless proxy signature scheme in the standard model," *Journal of Internet Technology*, vol. 23, no. 2, pp. 279–288, 2022.

[30] L. He, J. Ma, L. Shen, and D. Wei, "Certificateless designated verifier proxy signature scheme for unmanned aerial vehicle networks," *Science China Information Sciences*, vol. 64, pp. 1–15, 2021.

[31] R. Patil and Y. H. Patil, "A secure and efficient identity based proxy signcryption scheme for smart grid network." *Journal of Engineering Science & Technology Review*, vol. 15, no. 4, 2022.