World Wide Web (2023) 26:1599–1624
https://doi.org/10.1007/s11280-022-01106-2

EBSS: A secure blockchain-based sharing scheme for real estate financial credentials

Yadi Wu[1] · Guiyao Tie[1] · Yao Yu[1] · Jianxin Li[2] · Jun Song[1]

Received: 1 February 2022 / Revised: 26 May 2022 / Accepted: 19 September 2022 /
Published online: 4 October 2022

Abstract
With the development of Internet finance, the real estate financial credentials have been extensively utilized in capital and trading markets. The extensive use of these private credentials, e.g., exchange, circulation or sharing on public clouds, can cause severe security and privacy problems. Existing security solutions for financial credentials may either introduce undesired overhead or fail to provide traceability and anonymity for data sharing. In this paper, we propose an enhanced blockchain-based secure sharing scheme for real estate financial credentials, providing the following three properties: credential confidentiality, anonymous authentication, identity tracking and transaction auditing. A comprehensive evaluation, including security analysis, efficiency analysis and simulation evaluation, is presented to show the security and feasibility of the proposed scheme.

Keywords Blockchain · InterPlanetary File System (IPFS) · Real estate · Finance credential · Secure sharing

Guiyao Tie, Yao Yu and Jianxin Li contributed equally to this work.

This article belongs to the Topical Collection: *Web-based Intelligent Financial Services*
Guest Editors: Hong-Ning Dai, Xiaohui Haoran, and Miguel Martinez

✉ Jun Song
songjun@cug.edu.cn

Yadi Wu
wyd@cug.edu.cn

Guiyao Tie
tgy@cug.edu.cn

Yao Yu
yuyao@cug.edu.cn

Jianxin Li
jianxin.li@deakin.edu.au

[1] School of Computer Science, China University of Geosciences, Wuhan, Jincheng Street, Wuhan 430078, Hubei, China

[2] School of Information Technology, Deakin University, Brougham Street, Geelong VIC 3220, Victoria, Australia

1 Introduction

Real estate is regarded as a primary economic asset of human society [1]. Real estate financial credentials have been extensively used in domains such as capital market supervision, modern urban management, financial risk prevention, and so on [1, 2]. Since they involve the vital interests of the public, the management and utilization of real estate financial credentials have attracted close attention in recent years. With the development of Internet finance, it is necessary to establish standardized and unified real estate credential sharing mechanisms. Such a mechanism can not only improve the benefits of utilizing resources and assets, but also better protect the legal property interests of rights holders [3, 4].

However, from privacy and security perspectives, the sharing of financial credentials can raise critical issues [5, 6]. Some malicious behaviors are threatening the security of real estate financial credentials, such as theft, tampering, and forgery. Moreover, real estate financial credentials information, e.g., property information, rights status, and transaction records, is highly privacy and security sensitive. Without proper protection, serious security problems may occur.

Although the real estate financial credential is very security sensitive, most of current work on credential security has neglected to protect it efficiently. With respect to credential sharing, centralized storage architectures are widely adopted, and much of such work [7–10] focuses on data confidentiality and identity anonymity. Such an architecture, as widely known, leads to some prominent problems [11]. For example, ideally holders should be in charge of these credentials, but in most cases holders have to rely on third parties, such as banks or agents, to store, verify and validate their own credentials. On the other hand, some recent studies [5, 12, 13] have taken the secure credential sharing into consideration with the help of blockchain techniques. However, due to their public-transparent nature, such studies do not provide properties regarding circulation controllability and privacy-preserving.

To address above concerns, in this paper, we proposed an enhanced blockchain-based secure sharing scheme EBSS, particularly for real estate financial credentials. The main work and contributions of this paper include the following three aspects:

(1) Our proposed scheme can provide confidentiality of real estate credentials. In this paper, we adopt hierarchical encryption and access control approaches, which together enable credential confidentiality storage and distinct privileges management. The protection of shared credentials, thus enhancing the confidentiality and integrity of credentials, is one of the key feature that distinguishes our design from others.
(2) Our proposed scheme can achieve anonymous authentication. Authentication and commitment are the essential protection mechanisms for various security properties, i.e., integrity, authenticity, non-repudiation, and perfect hiding. Considering the strong requirements of identity privacy and transaction secrecy in blockchain financial services [14], anonymity is another essential security property that should be provided. In this paper, we adopt a combined group signature-based and commitment-based scheme to achieve anonymous authentication.
(3) Our proposed scheme can support identity tracking and transaction auditing. To realize secure sharing of real estate finance credentials, it is desirable for users to ensure the authenticity and verifiability of credential content. We adopt a distributed ledger mechanism to record the origin of the credentials, as well as all variations during

their storage and transmission processes. As evidence for tracking and auditing, these records are used in the consensus mechanism to improve auditing efficiency.

A comprehensive evaluation of the proposed scheme is provided. We first analyze the security of our design and then evaluate the simulation performance of the system implementation. Two general-purpose virtual simulators, i.e., Hyper ledger fabric and IPFS [15], are used in this work. We also compare our work with other related studies [16–19]. These evaluations show the effectiveness and feasibility of the proposed scheme. In addition, our scheme can be applied to many secure sharing scenarios, e.g., tickets management, accountability credentials, and private medical records.

The rest of this paper is outlined as follows. Related work is shown in Section 2. Section 3 briefly introduces security goals and threats, cryptographic tools, and the system model. Section 4 gives the detailed description of the proposed scheme. The security and performance evaluations are presented in Sections 5 and 6, respectively. The remaining issues and future work on the scheme are discussed in Section 7. Section 8 concludes this paper.

2 Related work

The security of real estate financial credentials has been a hot topic in recent years. Mani et al. [20] gave a comprehensive introduction on the security threats, awareness and risk management criteria for the existing security threats to real estate registration data. Kyle et al. [7] presented a physical network layout for preventing unauthorized access to real estate registration data. Deepa et al. [21] analyzed the potential threats to real estate registration information, especially for data storage and transmission. Kalia et al. [10] proposed a hybrid model of two-phase encoding with additive random noise value, which can protect private and sensitive information from leakage. Raymond et al. [22] proposed a threat assessment model, based on protective motive theory (PMT), for real estate information.

In terms of blockchain data sharing and protection, Liu et al. [23] proposed a blockchain-based and cloud-based traffic data sharing protocol, which can provide the security properties of anonymity and traceability. Li et al. [24] proposed a trusted big data sharing model based on blockchain and smart contracts. It can ensure the secure circulation of data resources. Several blockchain-based application schemes have been proposed, including data sharing model [25], real estate tokenization [26], and secure real estate transactions [5], respectively. None of such solutions provide traceability and anonymity of shared data, making it possible for attackers to compromise users' privacy. Nyaletey et al. [15] proposed the BlockIPFS system for evidence collection and trusted data traceability. But this work does not consider the anonymity of user identity. Zhang et al. [27] proposed a group signature and authentication scheme for blockchain-based mobile-edge computing. This scheme is able to verify the validity of the block and provide anonymity for both parties in a transaction. Since signatures and verifications cannot be performed independently in the blockchain, it is generally not appropriate for the case of changing group membership. Hong et al. [16] proposed a blockchain-based scheme for secure and accountable data transmission. This scheme is difficult to resist typical attacks such as impersonation attacks, man-in-the-middle attacks, and replay attacks. Fan et al. [19] proposed an improved ID-based signature authentication and blockchain-based secure data transmission scheme, which can provide the properties of authenticity, integrality and confidentiality of information;

however, this scheme does not take user privacy protection into consideration and the user's identity may be compromised.

To ensure entity authenticity during communication, many related solutions have been proposed in recent years. Harbi et al. [17] proposed an ECC-based enhanced authentication scheme, which can resist known security attacks, e.g., replay attack, denial-of-service attack, and impersonation attack. Jia et al. [18] presented a three-way authentication key protocol based on bilinear pairing. This protocol only requires one round communication to achieve mutual authentication and key agreement, while enabling user anonymity and untraceability. Chen et al. [28] proposed a threshold anonymous authentication (TAA) scheme in which users can obtain anonymous certificates without revealing private information. TAA is adopted in our paper for anonymous authentication.

Inspired by the previous work [15, 27], we proposed a blockchain-based secure sharing scheme for real estate credentials, integrating techniques such as blockchain, IPFS, and group signature. Moreover, our proposed scheme can provide the following advanced security properties, such as tamper-resistance of real estate credentials, traceability of credential circulation, and anonymity of user identities.

3 Preliminaries

3.1 Security goals and threats

In this section, we briefly introduce the potential threats and security goals, on which we focus.

1) Potential Threats: The threats to real estate financial credential sharing involve multiple aspects, e.g., caused by external or internal attacks. Our scheme design focuses on the following four aspects.

- **Credential forging/cheating**: Attackers may either send fake credentials for malicious purposes, or may use fake identities to send malicious information, e.g., sharing phishing links or spreading fake credentials.
- **Credential modification/replacing**: Attackers may modify, corrupt, or replace legitimate credentials, causing users to be misled or sharing to be invalid.
- **Credential plagiarizing**: Attackers may plagiarize other users' credentials or related private data. Such behaviors may compromise the confidentiality of these credentials or private data.
- **Correlation analysis**: Attackers may not know the true content of credentials. But they can obtain other related sensitive information by analyzing such credentials or private data, e.g., the user's location, the number of communications, or the length of the credential.

2) Security Goals: In this work, we consider the natural security features of blockchain and achieve the following security goals.

- **Authentication**: Before the system provides the service, the user's identity must be verified and authorized by a CA. Authentication can also be implemented between different users.

- **Data tamper resistance**: The system or user should be able to detect the modification or corruption of credentials. It may be caused by intentional or accidental factors during the credential transmission process.
- **Data confidentiality**: The secret data can only be revealed to authorized individuals, institutions or systems within a specific verification scenario.
- **Non-repudiation**: Any user cannot deny their past behaviors, such as signing, uploading or accessing credentials.
- **Anonymity**: During any information exchange process, a user's true identity should not be revealed so as to avoid illegal exploitation or impersonation.
- **Traceability**: All important information related to credentials should be traceable, e.g., data sources or historical operations. This means that the system should keep records of alterations and circulation.

3.2 Cryptographic tools

Cryptographic tools are the fundamental building blocks for security scheme designs. Here, we briefly introduce the cryptographic tools used in our scheme.

1) Blockchain: A blockchain is essentially a distributed ledger database for recording transactions. The ledger in a blockchain is maintained by all network nodes together. It is a transparent and irreversible process to set up the ledger. The format of transactions recorded in the blockchain can be customized as required. In this paper, we use the distributed ledger to record the transactions between the network participants. Such network participants can update the records in the ledger according to the consensus mechanism. The transaction information to be recorded is defined as TB, including the file level f_{level}, file authenticity f_{auc}, block number BN, file serial number f_{SN}. Based on smart contracts in the blockchain, some predefined rules and terms can be automatically executed [29]. Without the participation of a third party, smart contracts can also support trusted transactions. Thus, such transactions are traceable and irreversible.

Blockchain applications are usually divided into three categories [30], including public blockchain, private blockchain, and consortium blockchain. A consortium-based blockchain is adopted by our scheme. Each node of such a consortium blockchain is usually affiliated with a specific entity or organization, such as banks, core companies, suppliers, regulators, and so on. Only authorized users can join this blockchain. In this paper, the consortium blockchain is used to verify the legitimacy of transactions and to record historical data. It also provides features for security auditing and authority confirmation.

2) Interplanetary File System (IPFS): IPFS is a flexible peer-to-peer file system [31] that facilitates the storage and sharing of large files. It supports content addressing, allowing users to quickly access and verify data. An independent hash value Content-ID (CID) is generated based on the file content to identify the file, so only one file with the same content exists in the system and thus storage space is significantly reduced.

Since the blockchain is initially designed as a public ledger for recording transactions, most of the existing blockchain systems are more appropriate to the smaller transaction sizes for ensuring network performance. Therefore, it is either infeasible or expensive to store big files directly in a blockchain system. To address this issue, in this paper, we combine IPFS with blockchain to effectively alleviate the performance bottleneck of big file storage. Specifically, big files uploaded by users will be stored in IPFS, while their file addresses are reserved in the blockchain. Users can retrieve the corresponding files from IPFS according to the reserved addresses.

3) Group Signature: The group signature scheme can provide anonymous authentication for group members [32]. Different from other anonymity techniques, group signatures can relieve the burdens, e.g., distribution and verification of public key certificates. In particular, as an identity verification scheme, group signatures can also provide the security properties of message integrity and non-repudiation. A general group signature algorithm involves the following five steps: (1) *Setup:* After inputting the system parameters k, the group manager generates the group public key pk_{GM} and his own private key sk_{GM}. (2) *Join:* The group manager issues a group membership certificate *Cert* for the new user. (3) *Sign:* For a given message m, the user generates an anonymous signature σ_m using his own certificate $Cert_i$ and private key sk_{u_i}. (4) *Verify:* The verifier can use the group public key pk_{GM} to verify the signature σ_m. Such signature is acceptable if σ_m is a group signature on message m generated by a legitimate group member. (5) *Open:* The group manager can perform the *Open* algorithm to trace group signature σ_m.

In this paper, we adopt an effective group signature scheme as our anonymous authentication. It is a bilinear-map-based threshold anonymous announcement scheme TAA [28]. In terms of implementation features, it is essentially a group signature scheme, which can provide properties such as non-repudiation, anonymity, distinguishability and auditability. Such a scheme can achieve a better balance between hardware and software performance, thereby being adopted by Trusted Computing Group. In this paper, we tailor TAA specifically for blockchain services. Each legitimate user can obtain a valid identity credential through the *Setup* and *Join* algorithms. Anonymous signatures and message verification can be provided with the *Sign* and *Verify* algorithms.

4) Commitment: A commitment scheme is a two-party interactive protocol between a committer C and a receiver R. It consists of two main phases, i.e., the commitment phase and the open phase. In the commitment phase, the committer C sends a commitment *comm* with a secret message m to R. The commitment *comm* to m is generated with a parameter *open* and computation $comm = Com(m,open)$. With regard to the open phase, the committer C sends the original message m with the parameter *open* to R. Then R can verify that the committed message in *comm* is indeed m. A commitment scheme usually has two essential properties, i.e., hiding and binding.

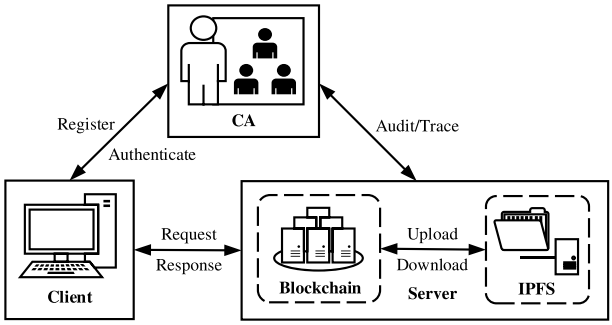
In this paper, a Pedersen commitment scheme [33] is adopted, which is based on the discrete logarithm problem (DLP). In the Pedersen commitment, the parameter *open* is set to $y \in \mathbb{Z}_q$. The commitment for message m can be defined as $comm = Com(m,y) = g^m h^y$, where g and h are security parameters. The receiver R can recompute $comm'$ with m, y and check whether $comm'$ is equal to received *comm*. The Pedersen commitment scheme has perfect hiding and computational binding properties. The commitment *comm* does not reveal any information about the message m. The success probability of any malicious committer finding another message with the same commitment is negligible. The receiver can be sure that m is the only message corresponding to the commitment.

3.3 Scheme model

The system model considered in this paper consists of three entities, as shown in Figure 1.

- **Certificate Authority (CA)**: CA is in charge of the registration of servers and users, generating public security parameters, distributing keys and issuing anonymous certificates, and tracing transaction disputes, and so on. In this model, CA also serves as a group manager of group signatures and is trusted by other entities.

Figure 1 System model



- **Server**: The server mainly provides request verification, file uploading and downloading, and data storage services. There are two types of trusted servers involved in our model, i.e., IPFS server and blockchain server. IPFS server is in charge of storing copy credentials and providing file indexing; blockchain server is able to verify requests and perform requested operations, e.g., file uploading, updating, and accessing. To facilitate later retrieval, tracks of such file operations will be recorded by the blockchain server.
- **Client**: Each client or user has a unique identity authenticated by CA. Clients can invoke requests to the server for services. Such requests have to be verified before responding. In this model, the client may be a malicious entity and may launch an active attack on the real estate credentials.

4 Scheme design

In this section, we provide a detailed design of EBSS. In this scheme, security mechanisms, such as anonymous authentication, hierarchical encryption, access control, and IPFS with blockchain, are integrated with the blockchain to ensure the security of real estate credential storage and sharing. Any data transmission between the user and the server follows the protocol flow shown in Figure 2. There are four main algorithms involved in data transmission: Request Generation (Algorithm 3), Hierarchical Encryption (Algorithm 4), Transaction Submission (Algorithm 5), and Access Control (Algorithm 6).

Anonymous authentication is mainly provided in Sign and Verify algorithms, i.e., Algorithms 1 and 2, respectively. For security concerns, each message has to be signed by the sender before being sent. The server will automatically drop messages that fail to be verified. To achieve confidentiality of data transmission, the commitment and hierarchical encryption techniques are embedded in Algorithm 3. In addition, distributed ledger and consensus mechanisms are adopted to perform identity traceability and data auditability, i.e., Algorithm 7. The properties of signature, commitment verification and identity tracking are also provided by Algorithm 7.

4.1 Protocol setup

The notations of our scheme are listed in Table 1. The specific cryptographic setup based on finite fields is as follows. First, three cyclic groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are chosen, all of which have sufficiently large prime order q. Two random generators $\mathbb{G}_1 = \langle P_1 \rangle$ and $\mathbb{G}_2 = \langle P_2 \rangle$ are selected along with a pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. The pairing $\hat{e}$ is a map from

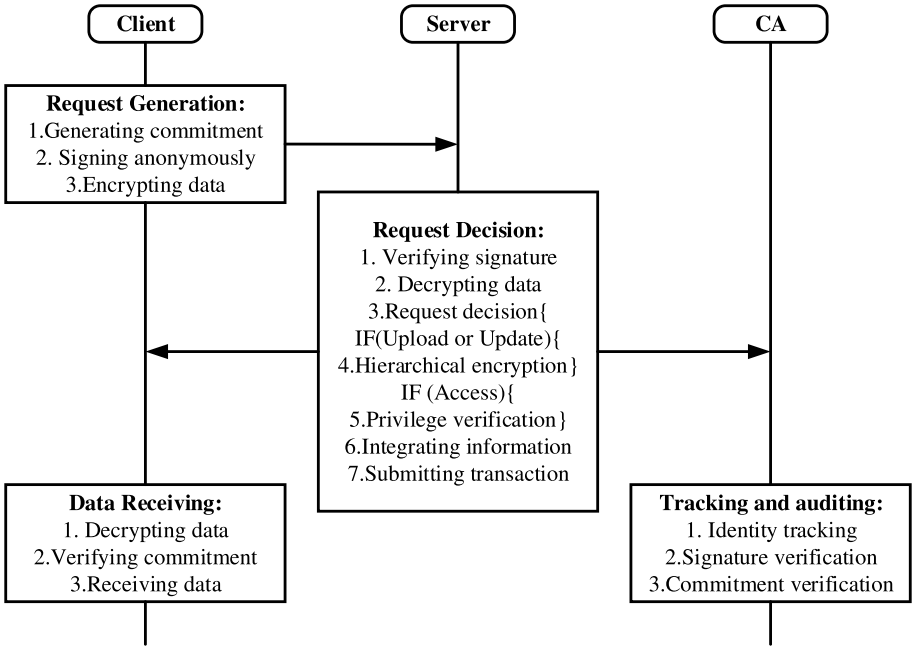


Figure 2 Protocol flow

Table 1 Notations

Notation	Explanation
$\mathbb{G}_1, \mathbb{G}_2$	Two additive cyclic groups with order q
$\mathbb{G}_T$	A multiplicative cyclic groups with order q
$\mathbb{Z}_q$	An additive integer cyclic group with order q
P_1,P_2	Generator for $\mathbb{G}_1$ and $\mathbb{G}_2$
(pk,sk)	A key pair: (public key, private key)
ct	A ciphertext encrypted with key pair
$\tilde{ct}$	A ciphertext encrypted with symmetric key
h_T	A transaction hash in blockchain
rq,msg	Request and message need to be sent or received
n_T	A timestamp
$comm$	A commitment result
key,k_{level}	A hierarchical symmetric key and its key level
f,h_f	A file and file hash, respectively
f_{level},u_{level}	The file level and user level, respectively
f_{SN},f_{auc}	A file serial number and file authenticity label, respectively
E_k,D_k	Encryption and decryption with key pair
Enc,Dec	Encryption and decryption with symmetric key

Algorithm 1 Sign.

procedure SIGN(Message msg, User level u_{level})
  $a \leftarrow Z_q$; $z \leftarrow Z_q$; $r \leftarrow Z_q$
  $\alpha \leftarrow H_2(u_{level})$; $\beta \leftarrow H_2(msg)$; $K = \lambda \cdot \alpha$; $L \leftarrow z \cdot \alpha$
  $R \leftarrow a \cdot A$; $S \leftarrow a \cdot B$; $T \leftarrow a \cdot C$; $\tau \leftarrow \hat{e}(S, X)^z$
  $\delta \leftarrow H_1(R \parallel S \parallel T \parallel \tau \parallel \alpha \parallel \beta \parallel K \parallel L \parallel n_T \parallel u_{level} \parallel msg)$
  $s \leftarrow z + \delta \cdot \lambda \pmod q$
  if Version 1 **then**
    $\sigma \leftarrow (R, S, T, \alpha, \beta, K, \delta, s, n_T)$
  else if Version 2 **then**
    $M = \lambda \cdot \beta$; $N \leftarrow r \cdot \beta$; $\nu \leftarrow \hat{e}(S, X)^r$
    $\theta \leftarrow H_1(R \parallel S \parallel T \parallel \tau \parallel \nu \parallel \alpha \parallel \beta \parallel K \parallel L \parallel M \parallel N \parallel n_T \parallel u_{level} \parallel msg)$
    $t \leftarrow r - \theta \cdot \lambda \pmod q$
    $\sigma \leftarrow (R, S, T, \alpha, \beta, K, M, \delta, \theta, s, t, n_T)$
  end if
  return σ
end procedure

$\mathbb{G}_1 \times \mathbb{G}_2$ to $\mathbb{G}_T$, which is bilinear, nondegenerate, and computable [34]. Two hash functions, i.e., $H_1 : \{0,1\}^* \rightarrow \mathbb{Z}_q$ and $H_2 : \{0,1\}^* \rightarrow \mathbb{G}_1$, are chosen to map an arbitrary-length binary string to an integer and a $\mathbb{G}_1$ element, respectively.

Second, CA issues a certificate for each user to ensure valid authentication. CA holds a secret key (x,y) and a public key (X,Y), where $x, y \leftarrow \mathbb{Z}_q, X = x \cdot P_2 \in \mathbb{G}_2, Y = y \cdot P_2 \in \mathbb{G}_2$. The certificate ($A$,$B$,$C$) is a triplet and is constructed with the true identity $\lambda \in \mathbb{Z}_q$ of the user and the secret key (x,y) of CA. To be specific, $A \leftarrow \gamma \cdot P_1, B \leftarrow y \cdot A$, and $C \leftarrow (x \cdot A + \lambda xy \cdot A)$, where γ is a random number chosen from $\mathbb{Z}_q$. Therefore, such a certificate is unique for a specific user and is anonymous for other users.

Third, we adopt the Role-Based Access Control (RBAC) model and the "principle of least privilege" to achieve system access control and resource secure sharing [35]. Therefore, during user registration and file storage processes, the trusted server will construct a User-Role-Privilege (URP) table, so as to form a one-to-many mapping between user level u_{level}, file level f_{level}, and hierarchical key level k_{level}. To facilitate fine-grained management, the server keeps this URP table and assigns corresponding hierarchical keys by file level f_{level}; and restricts access to files depending on user level u_{level}. Furthermore, such access control can keep consistency among user level u_{level}, file level f_{level} and key level k_{level}.

4.2 Signing and verification

A group signature scheme is used to achieve anonymous authentication. All messages, both incoming and outgoing, need to be authenticated. It means that every sent message should be signed first, while every received message has to verify the signature first. In this paper, we deploy a group signature scheme similar to TAA [28].

Algorithm 1 signs on the message and generates a signature σ. In this algorithm, the triplet (R,S,T) is an anonymous certificate generated by shuffling (A,B,C). On the basis of anonymous certificates, each message is signed. The calculation of δ, s, θ and t provide the correlation credential between the anonymous certificate and the user's true identity λ.

Algorithm 2 Verify.

1: **procedure** VERIFY(Message msg, User level u_{level}, Signature σ)
2: **if** n_T has expired or $\alpha \neq H_2(u_{level})$ or $\beta \neq H_2(msg)$ or $e(R, Y) \neq e(S, P_2)$ **then**
3: **return** Reject
4: **end if**
5: $\rho'_a \leftarrow \hat{e}(R, X)$; $\rho'_b \leftarrow \hat{e}(S, X)$; $\rho'_c \leftarrow \hat{e}(T, P_2)$
6: $\tau' \leftarrow (\rho'_b)^s \cdot (\rho'_c/\rho'_a)^{-\delta}$
7: $L' \leftarrow s \cdot \alpha - \delta \cdot K$
8: **if** Version 1 and $\delta = H_1(R \parallel S \parallel T \parallel \tau' \parallel \alpha \parallel \beta \parallel K \parallel L' \parallel n_T \parallel u_{level} \parallel msg)$ **then**
9: **return** Accept
10: **else if** Version 2 **then**
11: $\nu' \leftarrow (\rho'_b)^t \cdot (\rho'_a/\rho'_c)^{-\theta}$; $N' \leftarrow t \cdot \beta + \theta \cdot M$
12: **if** $\theta = H_1(R \parallel S \parallel T \parallel \tau' \parallel \nu' \parallel \alpha \parallel \beta \parallel K \parallel L' \parallel M \parallel N' \parallel n_T \parallel u_{level} \parallel msg)$ **then**
13: **return** Accept
14: **end if**
15: **end if**
16: **return** Reject
17: **end procedure**

They are based on bilinear pairing and modulo algebra, respectively. To resist the replay attack, a timestamp n_T is embedded in the message signature σ. During the signature and verification process, there are two versions of Sign and Verify algorithms, respectively. Version 1 is for normal usage, and Version 2 focuses on applications with high security requirements, where an enhanced security signature can be realized with the help of M, N and ν.

The verification of the signature is described in Algorithm 2. First, the validity of the timestamp n_T in the signature σ is verified, and such a check can help determine whether σ has been replayed. The integrity of the message is also provided by checking whether $\alpha \neq H_2(u_{level})$ and $\beta \neq H_2(msg)$, so that any corruption of the message can be detected. Then, a comparison of $\hat{e}(R, Y)$ and $\hat{e}(S, P_2)$ is performed so as to check the internal relationship of R, S and Y, i.e., $S = a \cdot B = ay \cdot A = y \cdot R$. Thus, $\hat{e}(R, Y) = \hat{e}(A, P_2)^{ay} \equiv \hat{e}(S, P_2)$. If the signature σ and the message msg are successfully transmitted without any corruption, τ should be equivalent to τ' and ν' should also be equivalent to ν. The correctness is as follows: $\tau' = (\rho_b)^s \cdot (\rho_c/\rho_a)^{-\delta} = \hat{e}(S, X)^s \cdot \hat{e}(T, P_2)^{-\delta} \cdot \hat{e}(R, X)^{\delta} = \hat{e}(S, X)^z \equiv \tau$. Similarly, the equivalence of ν' and ν can be verified in this way. Furthermore, if msg, u_{level} and σ are also successfully transmitted, the correctness of δ and θ can be verified, as shown in line 7 and 12.

4.3 Secure transmission

The data transmission procedure is shown in Figure 2. Before data transmission, the user first generates a request message, i.e., rq in Algorithm 3, asking the server for a specific secure transmission. In case of an *Upload* request or an *Update* request, the server will perform a hierarchical encryption in Algorithm 4 to provide data confidentiality. These

Algorithm 3 Request generation.

1: **procedure** REQUEST(Data $data$)
2: $d \leftarrow \mathbb{Z}_q; h_f \leftarrow H_1(f)$
3: **if** $Upload$ **then**
4: $comm \leftarrow Com(h_f, d); ct = E_{pk_{ser}}(f \parallel d)$
5: $msg.data = ct \parallel f_{level} \parallel comm$
6: **else if** $Update$ **then**
7: $comm \leftarrow Com(h_f, d); ct = E_{pk_{ser}}(f \parallel d)$
8: $msg.data = ct \parallel h_T \parallel comm$
9: **else if** $Access$ **then**
10: $comm \leftarrow Com(h_T, d); ct = E_{pk_{ser}}(h_T \parallel d)$
11: $msg.data = ct \parallel h_T \parallel comm$
12: **end if**
13: $\sigma \leftarrow Sign(msg, u_{level})$
14: **return** $rq \leftarrow (msg, u_{level}, \sigma)$
15: **end procedure**

encrypted data will be stored in IPFS and their operation tracks are recorded on the blockchain, as shown in Algorithm 5, so that the server can implement related security audits and data traceability later. If it is an *Access* request, the server will execute the Access Control algorithm, as shown in Algorithm 6, to make a decision whether or not the protected data can be accessed. For different transmission requests, the receivers should verify their validity and each transmitted file should be encrypted, thereby ensuring confidentiality and integrity for the following data transmissions.

1) Secure request: Request generation procedure is shown in Algorithm 3. Such a request provides two main security properties: secure commitment and message confidentiality. $Com(h_f,d)$ denotes a commitment transformation for the file hash h_f, where d is a random parameter and *comm* is a commitment. Similarly, $Com(h_T,d)$ refers to a commitment transformation of the transaction hash h_T. To preserve confidentiality, the protected data, e.g., the file f, the transaction hash h_T, and the parameter d, should be encrypted using the public key pk_{ser} to generate the ciphertext ct, i.e., lines 4, 7, and 10. For three specific requests, the algorithm generates one corresponding message msg, i.e., lines 5, 8, and 11, respectively. These messages consist of different elements, including ciphertext ct, file level f_{level}, transaction hash h_T, and commitment $comm$. Finally, the message msg, user level u_{level}, and signature σ are sent back to the user as a request, i.e., lines 13-14.

2) Hierarchical encryption: Upon receiving an *Upload* or *Update* request, the server performs the hierarchical encryption algorithm as a response, i.e., Algorithm 4. Such hierarchical encryption mainly provides confidentiality for file storage. The algorithm, based on the received message msg, first derives the ciphertext ct, which can be decrypted to obtain the plaintext f, i.e., line 5. Then, the server makes a decision with respect to the request. For the *Upload* request, the server selects the corresponding hierarchical key key, i.e., line 7, which is consistent with the requested file level $rq.f_{level}$; however, for an *Update* request, the server, along with the transaction hash $rq.h_T$, retrieves the transaction record TR, and finds the appropriate hierarchical key key, i.e., line 9-10. Such a key level k_{level} depends on the file level f_{level} in TR. It should be noted that a query function *Query* is adopted to achieve information retrieval, i.e., line 9,

Algorithm 4 Hierarchical encryption.

Algorithm 4 Hierarchical encryption.

1: **procedure** ENCRYPTION(Request rq)
2: **if** $Verify(rq)$ fails **then**
3: **return** Ignore
4: **end if**
5: $f \leftarrow D_{sk_{ser}}(rq.ct)$
6: **if** rq is $Upload$ **then**
7: $k_{level} \leftarrow rq.f_{level}$
8: **else if** rq is $Update$ **then**
9: $TR \leftarrow Query(rq.h_T)$
10: $k_{level} \leftarrow TR.f_{level}$
11: **end if**
12: $\tilde{ct} \leftarrow Enc(key, f)$
13: **return** $\tilde{ct}$
14: **end procedure**

Algorithm 5 Transaction submission.

1: **procedure** SUBMISSION(CID, signature σ, commitment $comm$)
2: $f_{auc} \leftarrow CID \parallel \sigma \parallel comm \parallel d$
3: $TB.f_{level} \leftarrow f_{level}$
4: $TB.f_{auc} \leftarrow f_{auc}$
5: $TB.f_{SN} \leftarrow f_{SN}$
6: $TB.BN \leftarrow$ `blockchain.`BN
7: $h_T =$`blockchain.`$hash(TB)$
8: **return** h_T
9: **end procedure**

particularly for file updating, access control and data tracking. For the purpose of ensuring data confidentiality, the algorithm encrypts the protected file f using the selected *key*, i.e., lines 12-13, so that a new ciphertext $\tilde{ct}$ is generated and stored in the server. In addition, *TR* retrieved by the $Query(h_T)$ function refers to a transaction record on the blockchain. To be specific, a *TR* at least consists of transaction hash h_T, file level f_{level}, block number *BN*, file authenticity f_{auc}, file serial number f_{SN}, and so on. f_{SN} is generated by the server. Each file has a unique f_{SN}, even if it is a different version of this file.

3) Transaction information integrity: To ensure that $\tilde{ct}$ is not tampered with, the transaction information has to be stored in the blockchain, e.g., the file storage address *CID*, the signature σ, and the commitment *comm*. Such transaction information provides proof for later retrieval and verification of $\tilde{ct}$. Algorithm 5 shows the process of transaction submission. Such an algorithm mainly keeps the traceability and integrity of the transaction information. The server needs to create an object *TB* for each transaction, which is used to integrate the transaction information. Typically, a *TB* contains elements such as file level f_{level}, file authenticity f_{auc}, block number *BN*, and file serial number f_{SN}. The information integration process in the algorithm is shown in lines 3-7. Here, blockchain.*BN* means the block number *BN*, which is obtained by calling the inline function of the blockchain. After the transaction information is integrated, the server submits the

Algorithm 6 Access control.

1: **procedure** ACCESSCONTROL (Request rq)
2: **if** $Verify(rq)$ fails or $rq.comm \neq Com(rq.h_T, d \leftarrow D_{sk_{ser}}(rq.ct))$ **then**
3: **return** Ignore
4: **end if**
5: $f_{level}, f_{auc} \leftarrow Query(rq.h_T)$
6: $u_{level} \leftarrow Query(rq.\sigma.T)$
7: **if** $u_{level} \geq f_{level}$ **then**
8: $\tilde{ct} \leftarrow \texttt{IPFS}(f_{auc}.CID)$
9: $k_{level} \leftarrow f_{level}$
10: $f \leftarrow Dec(key, \tilde{ct}); h_f \leftarrow H_1(f)$
11: $comm' \leftarrow Com(h_f, f_{auc}.d)$
12: **if** $comm' \neq f_{auc}.comm$ **then**
13: **return** Ignore
14: **end if**
15: **return** $f, f_{auc}.\sigma$
16: **else**
17: **return** Ignore
18: **end if**
19: **end procedure**

transaction object *TB* to the blockchain and generates the corresponding transaction hash h_T by blockchain.*hash*(*TB*), i.e., line 7. Once verified and validated by the consensus mechanism, such a transaction will be permanently recorded in the blockchain, i.e., line 8.

4) Access privilege verification: To achieve effective access control and data sharing, the server performs the Access Control algorithm, i.e., Algorithm 6, when receiving an *Access* request. With the basis of h_T, the algorithm first invokes the *Query* function to derive f_{level} and f_{auc}, i.e., line 5. Then, relying on the anonymous certificate contained in the user signature σ, the algorithm invokes *Query*($rq.\sigma.T$) to find the user level u_{level}, i.e., line 6. Next, the server makes an access control decision by comparing u_{level} with f_{level}. If u_{level} is less than f_{level}, it means that the user does not have enough privileges to access the file; otherwise, the access is permissible. The server will extract the ciphertext $\tilde{ct}$ from IPFS and decrypt it into plaintext f, i.e., lines 8-11. To check if the file f has been tampered or corrupted, the server will generate a new commitment $comm'$ by computing $Com(h_f f_{auc}.d)$. If $comm'$ is equal to the verified $f_{auc}.comm$, it means that f is the correct one. Finally, the server will send the corresponding file f and signature σ to the user, i.e., line 15.

4.4 Traceability and auditability

For preserving anonymity concerns, it is not desired to track the true identity of the signer from the signature. Moreover, the tracking and auditing features are not necessary for legitimate users, signatures and commitments. In particular, our scheme introduces specialized security mechanisms, e.g., anonymous authentication, hierarchical encryption, security commitment, and access control, which can provide sufficient security. However, the abilities of tracking and auditing by CA will still be reserved for effective management purposes. Algorithm 7 describes the tracking and auditing processes, involving three different functions: Identity tracking, Signature

Algorithm 7 Tracking and auditing.

1: **function** IDENTITY TRACKING(σ)
2: **for** All λ in $\{\lambda_1, \lambda_2, \ldots\}$ **do**
3: **if** $\sigma.T == x \cdot \sigma.R + x \cdot \lambda \cdot \sigma.S$ **then**
4: The signature is traced to identity λ
5: **return** Found
6: **end if**
7: **end for**
8: **return** Not Found
9: **end function**
10: **function** SIGNATURE VERIFICATION(TR, rq)
11: $\sigma^{\dagger} \leftarrow Sign(rq.msg^{\dagger})$
12: **if** $\sigma^{\dagger} == rq.\sigma^{\dagger}$ and $\sigma^{\dagger} == TR.f^{\dagger}_{auc}.\sigma$ **then**
13: **return** Sign Correct
14: **end if**
15: **return** Sign Incorrect
16: **end function**
17: **function** COMMITMENT VERIFICATION(TR, rq)
18: $f^{\dagger}, d^{\dagger} \leftarrow D_{sk_{ser}}(rq.ct^{\dagger}); h^{\dagger}_{f} \leftarrow H_1(f^{\dagger})$
19: $comm^{\dagger} \leftarrow Com(h^{\dagger}_{f}, d^{\dagger})$
20: **if** $comm^{\dagger} == TR.f^{\dagger}_{auc}.comm$ **then**
21: **return** Com Correct
22: **end if**
23: **return** Com Incorrect
24: **end function**

verification and Commitment verification. Note that the transaction record *TR*, request *rq* and signature σ are reserved information in the distributed ledger. For the case of identity tracking, since the CA knows the true identity λ and the secret key x of each user, it can verify the user's certificate by matching the internal relationship of (R, T, S), i.e., $\sigma.T = a \cdot x \cdot \sigma.A + a \cdot \lambda xy \cdot \sigma.A = x \cdot \sigma.R + x \cdot \lambda \cdot \sigma.S$

For the Signature verification process, a new signature $\sigma^{\ddagger}$ can be generated based on the reserved request message ($rq.msg^{\ddagger}$). Then, CA is able to compare $\sigma^{\ddagger}$ with $rq.\sigma^{\ddagger}$ and $TR.f^{\dagger}_{auc}.\sigma$ to verify the correctness of the signature, i.e., lines 11-13. Once the Commitment verification function is executed, both a new $f^{\ddagger}$ and $d^{\ddagger}$ can be obtained by decrypting the previous request ciphertext ($rq.ct^{\ddagger}$). Then a new commitment $comm^{\ddagger}$ can be generated with such $h^{\dagger}_{f}$ and $d^{\ddagger}$, thus enabling the auditing of the commitment, i.e. lines 18-21. As long as the commitment $TR.f^{\dagger}_{auc}.comm$ stored by the server has not been tampered with or corrupted, it should hold that $comm^{\ddagger}$ is equivalent to $TR.f^{\dagger}_{auc}.comm$.

5 Security analysis

In this section, we analyze the security of the proposed scheme in five aspects, with emphasis on the main security features involved in the algorithm implementation.

5.1 Security of the signature

The security of the signature in this scheme is guaranteed by the hardness of the **bilinear LRSW assumption** [36]. Suppose that the $Setup(1^k)$ algorithm produces the three cyclic groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$, where two generators are P_1 and P_2, the order is q, and k is a security parameter. There exist $X, Y \in \mathbb{G}_2$, $X = x \cdot P_2$, and $Y = y \cdot P_2$. Let $\mathcal{O}_{X,Y}(\cdot)$ be an oracle with $\gamma, \lambda \in \mathbb{Z}_q$ as input and a triplet (A,B,C) as output, where $A \leftarrow \gamma \cdot P_1, B \leftarrow y \cdot A$, and $C \leftarrow (x \cdot A + \lambda xy \cdot A)$. For a probabilistic polynomial time (*p.p.t.*) adversary $\mathcal{A}$, it is impossible to construct the triplet (A,B,C) without knowing the secret key (x,y). $v(k)$ is a negligible function for all $\mathcal{A}$, defined as follows:

$$\begin{aligned} &Pr[(\mathbb{G}_1, \mathbb{G}_2, P_1, P_2, q, \hat{e}) \leftarrow Setup(1^k); x \leftarrow \mathbb{Z}_q; y \leftarrow \mathbb{Z}_q; \\ &X = x \cdot P_2; Y = y \cdot P_2; (\lambda, A, B, C) \leftarrow \mathcal{A}^{\mathcal{O}_{X,Y}}(\mathbb{G}_1, \mathbb{G}_2, P_1, P_2, q, \hat{e}) : \\ &f \in \mathbb{Z}_q \wedge A \in \mathbb{G}_1 \wedge B = y \cdot A \wedge C = (x \cdot A + \lambda xy \cdot A)] \leq v(k). \end{aligned} \tag{1}$$

This means that given the system public parameters $(\mathbb{G}_1, \mathbb{G}_2, P_1, P_2, q, \hat{e})$ and public key (X,Y), it is impossible for an adversary $\mathcal{A}$ to construct a triplet (A,B,C) without knowing the secret key (x,y), the random number γ, and the secret true identity λ. Only the CA with the secret key (x,y) can construct a valid certificate for the user, which can guarantee the effectiveness of our authentication scheme. Moreover, with regard to the Sign algorithm, signature σ is constructed with the secret parameters a, z and r, the secret identity λ, and the shuffled credential (R,S,T) where $R = a \cdot A$, $S = a \cdot B$, and $T = a \cdot C$. Any adversary cannot generate a valid anonymous signature σ without knowing (A,B,C) and λ [28]. Thus, the signed message cannot be forged or modified during the transmission processes, which can provide non-repudiation, authenticity, and integrity of the sent messages.

5.2 Security of identity anonymity

In our scheme, the anonymity of user identity involves four aspects: first, the true identity of the user λ and the secret key (x,y) of CA are encapsulated in the triplet credential (A,B,C), i.e., $C \leftarrow (x \cdot A + \lambda xy \cdot A)$. With the hardness of the **bilinear LRSW assumption** [37] mentioned above, it is impossible for a *p.p.t* adversary $\mathcal{A}$ to extract λ from C without knowing the private key (x,y). Second, the true identity λ is encapsulated into the credential computation of K, s and t. Due to the hardness of the **discrete logarithm problem (DLP)** [38], it is obvious that extracting the true identity λ from K, s and t is not possible. Third, since there is no isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$ in the asymmetric pairing setting, it is not feasible to link (R,S,T) with the (A,B,C) without knowing the secret parameter a [39]. This also provides anonymity of the user identity. Finally, the true identity of the user λ is independent of the transaction information TR recorded in the blockchain. Hence, no one except the CA can trace the true identity of the signer, and the anonymity of the user's identity can be effectively guaranteed.

5.3 Security of transaction records

As described in the hierarchical encryption Algorithm 4, two primitives h_T and f_{auc} are included in a transaction record of the blockchain, where h_T is a transaction hash and f_{auc}

is a file authenticity label. h_T changes with each record. In general, the security of a hash function depends on three underlying properties, i. e., collision resistance, preimage resistance, and second preimage resistance. In our scheme, we adopt SHA-2 algorithm, which works on the **Merkle-Damgåd structure** and whose soundness has been proved [40, 41]. On the other hand, there are two primitives, signature σ and commitment *comm*, for constructing f_{auc}. As mentioned earlier, the security of the signature σ depends on the hardness of the **bilinear LRSW assumption** [38]. Any authorized user can access the file f and compute the commitment $comm' = Com(h_f, d)$, so as to verify the integrity of f. We adopt a Pedersen commitment scheme [33], which can provide the **Perfect Hiding Property**. It is computationally infeasible for any *p.p.t* adversary $\mathcal{A}$ to extract h_f from the commitment *comm* without knowing the random number d. To be specific, assuming that an attacker has the ability to tamper with the commitment *comm*, there exists different d' and f' such that $g^{f'} h^{d'} \equiv g^f h^d (\mod p)$, where p and q are security parameters. For the attacker, it is equivalent to solving the **discrete logarithmic difficulty problem** [42]. Therefore, the assumption that the attacker can successfully tamper with the commitment is invalid.

5.4 Security of access control

We adopt access control technique to make decisions based on user level u_{level}. Due to the consistency between user level u_{level}, file level f_{level} and key level k_{level}, for an authorized user, the corresponding key level is used to decrypt the file f, i.e. $f \leftarrow Dec(key, \tilde{ct})$. Note that in EBSS only a trusted server holds the hierarchical symmetric encryption key *key*. Additionally, the cryptographic strength of hierarchical symmetric encryption depends on the properties of the underlying symmetric encryption scheme. In this paper, we use the AES algorithm, which is well-known as a standard for symmetric data encryption [43]. Much studies [44, 45] have proven the practical security of AES, e.g., resistance to brute force attacks, differential attacks, and square attacks. Therefore, the confidentiality of the access data can be provided.

Moreover, the computational constraint property of Pedersen commitment relies on the **discrete logarithm assumption (DLA)** [46]: $Pr[g^{m_1} h^{r_1} \ mod q]$-$Pr[g^{m_2} h^{r_2} \ mod q]| = 0$, where g, h are generators of the group $\mathbb{G}_q$, q is a large prime, $m_1, m_2, r_1, r_2 \in \mathbb{Z}_q$, and r_1,r_2 follow normal distribution. It means that, for any $m_1 \neq m_2$, there exists an $r_1 \neq r_2$ such that $Com(m_1,r_1) = comm = Com(m_2,r_2)$. In other words, such Pedersen commitment is *computationally binding* [47] since the committer is unable to open a commitment to m_1 as $m_1 = m_2$, unless the Discrete Logarithm Problem (DLP) can be solved. In EBSS, this assumption guarantees that any adversary can only tamper with the commitment *comm* with negligible probability. In other words, on the basis of Pedersen commitment, it is infeasible for any two different files f and f' to generate the same commitment value *comm*. Thus, the authenticity of the accessed data can be guaranteed in our scheme.

5.5 Security of confidential transmission

The data *data* is encrypted with CA's asymmetric key pk_{ser} before transmission, i.e., $ct = E_{pk_{ser}}(data \parallel d)$, where *data* may be a file f or a transaction hash h_T. In this scheme, we use the ElGamal encryption algorithm. The security of ElGamal encryption is guaranteed by the hardness of the **Decisional Diffie-Hellman (DDH) Assumption** [48]: Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be cyclic groups, q be a large prime order, P_1 and P_2 be two generators, and $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ be a pairing. Let $X, Y, Z \in \mathbb{G}_1$, $X = x \cdot P_1$, $Y = y \cdot P_1$, and $Z = z \cdot P_1$,

where x,y,z are randomly and independently chosen from $\mathbb{Z}_q$. Then given the group parameters $(\mathbb{G}_1, \mathbb{G}_2, P_1, P_2, q, \hat{e})$, for any probabilistic-polynomial time ($p.p.t$) algorithm $\mathcal{G}$, the advantage $\mathbf{Adv}^{DDH}_{\mathcal{A},\mathbb{G}_1}$ defined as follows is negligible:

$$\begin{aligned}\mathbf{Adv}^{DDH}_{\mathcal{A},\mathbb{G}_1} =& |Pr[x, y, z \leftarrow \mathbb{Z}_q; X = x \cdot P_1, Y = y \cdot P_1, Z = z \cdot P_1; \mathcal{G}(\mathbb{G}_1, \mathbb{G}_2, \\ & P_1, P_2, X, Y, Z, q) = 1] - Pr[x, y \leftarrow \mathbb{Z}_q; X = x \cdot P_1, Y = y \cdot P_1, \\ & Z = xy \cdot P_1; \mathcal{G}(\mathbb{G}_1, \mathbb{G}_2, P_1, P_2, X, Y, Z, q) = 1]|.\end{aligned} \tag{2}$$

Furthermore, both $< X = x \cdot P_1, Y = y \cdot P_1, Z = xy \cdot P_1 >$ and $< X = x \cdot P_1, Y = y \cdot P_1, Z = z \cdot P_1 >$ are computationally indistinguishable. It is equivalent to the semantic security in ElGamal encryption [39].

The **Perfect Hiding Property** is provided by the Pedersen commitment. It has been proved that, for any value r, the commitment is uniformly distributed and the parameters are chosen randomly and uniformly [42], i.e., $|Pr[Com(m_1, r)] - Pr[Com(m_2, r)]| = 0$, where $Com(m_1, r), Com(m_2, r) \in \mathbb{G}_q$, $m_1, m_2, r \in \mathbb{Z}_q$, and r follows the uniform distribution. In our scheme, the perfect hiding property guarantees that any adversary in the communication can only obtain data from rq with a negligible probability. Thus, in our scheme, *comm* can perfectly hide all the information about the *data*, including the file hash h_f and the transaction hash h_T. Thus, the confidential transmission of messages is provided in our scheme.

5.6 Security properties analysis

The security properties provided by EBSS are summarized in Table 2. Note that *Encryption* refers specifically to the hierarchical encryption and asymmetric encryption during the request process.

In the proposed scheme, *Signature* ensures that the signer cannot deny the signed data and other users can verify the correctness of the signature. The signature is accomplished by anonymous certificates and only CA can trace the true identity of the certificate. It further provides anonymity and traceability of user identity during system services. The proposed *Commitment* mechanism is perfect hiding and computationally binding, which can provide five security features for the transaction process, including confidentiality, integrity, verifiability, traceability, and resistance to collusion attacks. *Encryption* can ensure confidentiality and integrity during data transmission and storage. Moreover, the *timestamp* contained in the signature can prevent the transaction information from being replayed.

Table 2 Description of security properties

Security properties	Signature	Commitment	Encryption	Timestamp
Confidentiality		✓	✓	
Integrity		✓	✓	
Non-repudiation	✓			
Anonymity	✓			
Verifiability	✓	✓		
Traceability	✓	✓		
Collusion Attack resilience		✓		
Message Replay Attack resilience				✓

6 Performance evaluation

In this section, we evaluate the performance of the proposed scheme, which involves four aspects, i.e., computation overhead, communication overhead, simulation evaluation, and comparison with related schemes. We apply the proposed algorithms in simulations with real blockchain settings, where we show the feasibility and effectiveness of our scheme. All experiments are conducted on Ubuntu 18.04 LTS, with 2.8 GHz Intel $i5-8400$ CPU, 16 GB memory and 256 GB SSD.

6.1 Computation overhead

In our scheme, the introduced computation overhead is mainly related to two procedures, such as signing and verification and request generation.

For the signing and verification procedure, since all messages need to be signed before sending and verified after receiving, such a procedure will introduce some additional computational overhead. We use the existing implementation results in [49] as an evaluation reference, the operations with higher computation cost are the scalar multiplication in $\mathbb{G}_1$, the exponential operation in $\mathbb{G}_T$, and the pairing operation, respectively. In contrast, both the hash and arithmetic operations in $\mathbb{Z}_q$ introduce very little overhead. Owing to the bilinear property of the mapping $\hat{e}$, some exponential operations in $\mathbb{G}_T$ can be transformed into scalar multiplications in $\mathbb{G}_1$ for faster computation. On the basis of these preparations, the computation overheads of signing and verification are as follows: Sign v1, $7 \cdot \mathbb{G}_1 + 1 \cdot P$; Sign v2, $11 \cdot \mathbb{G}_1 + 2 \cdot P$; Verify v1, $4 \cdot \mathbb{G}_1 + 3 \cdot P$; Verify v2, $8 \cdot \mathbb{G}_1 + 6 \cdot P$. Here, $\cdot\mathbb{G}_1$ refers to a scalar multiplication on $\mathbb{G}_1$ and $\cdot\ P$ refers to a pairing operation. We evaluate these operations with the implementation results from [19]. On average, a scalar multiplication on $\mathbb{G}_1$ takes 2.165 ms while a bilinear pairing operation takes 5.427 ms. Hence, the computational overheads for signing and verifying are 20.582 ms (Sign v1), 34.669 ms (Sign v2), 24.941 ms (Verify v1), and 49.882 ms (Verify v2), respectively.

During the request generation procedure, i.e., Algorithm 3, the user needs to commit and encrypt the file f, as well as h_T. Compared to other operations, both the encryption and decryption operations take less time and thus are negligible [49]. The commitment computation may introduce some extra overhead. However, since such a commitment can be precalculated, it will not introduce extra computation overhead in practice. A commitment in this scheme involves two main computation overheads: the modular exponentiation M_e and the modular multiplication M_m. With the implementation results in [19], the module exponentiation operation takes approximately 0.339 ms and the modular multiplication operation takes approximately 0.001 ms. Therefore, the computational overhead required for a commitment operation is $2 \cdot M_e + 1 \cdot M_m$, which is 0.697 ms.

6.2 Communication overhead

In our scheme, the communication overhead introduced is mainly related to two procedures: authentication and request generation. We set an element length in $\mathbb{G}_1$ to be 1024 bits, i.e., $|\mathbb{G}_1| = 1024$ bits; an element length in $\mathbb{G}_2$ to be 2048 bits, i.e., $|\mathbb{G}_2| = 2048$ bits; and an element length in $\mathbb{Z}_q$ to be 160 bits, i.e., $|\mathbb{Z}_q| = 160$ bits. The timestamp n_T is 32 bits in length, i.e., $|n_T| = 32$ bits. Since the signature is included in each message, the

communication overhead of authentication is determined by the signature size. Specifically, the communication overhead introduced is approximately $6|\mathbb{G}_1| + 2|\mathbb{Z}_q| + |n_T| = 6.496$ Kb for version 1 and $7|\mathbb{G}_1| + 4|\mathbb{Z}_q| + |n_T| = 7.840$ Kb for version 2.

Each user request contains a signature. The communication overhead of the signature is given as in version 1 or in version 2 above. Besides the signature, there are also an asymmetrically encrypted ciphertext *ct*, a commitment *comm*, a file level f_{level} or a transaction hash h_T to be transmitted during such request procedure. In particular, the length of the asymmetric encrypted ciphertext *ct* depends on the data size that the user needs to upload. It is indispensable for every blockchain scheme. For example, such data size may be measured in Kb, Mb, or Gb in real applications. Thus, it is reasonable that such communication overhead is ignored in this scheme. A commitment *comm* in $\mathbb{Z}_q$ is 160 bits long, i.e., $|comm| = 160$ bits; a file level f_{level} or a transaction hash h_T, $f_{level}, h_T \in \mathbb{Z}_q$, is 160 bits long, i.e., $|f_{level}| = 160$ bits at most or $|h_T| = 160$ bits. We take these overheads into consideration, and the maximum communication overhead introduced is approximately 6.816 Kb for version 1 and 8.160 Kb for version 2.

6.3 Simulation evaluation

For a non-saturated network, since the introduced extra overheads are relatively small, it is hard to detect the performance impact of the proposed scheme. To better understand the impact of the security scheme, we measure the overall communication impact by increasing the data upload and download capacity, along with and without the security overhead for both versions. We use an abstract scenario to simulate the secure authentication process, with higher message generation counts and more intensive signing and verification. Such a simulation is only intended to help us better understand the impact of the security scheme under extreme conditions.

Our simulation is conducted on a common and open source blockchain simulator, i.e., Hyper ledger fabric 1.4.4 and IPFS 0.4.19 [15]. Such network simulation deployment can support image credentials storage and integrity verification without uploading all data to the blockchain. Our simulation dataset comes from "Zillow" [50], which is a public dataset on real estate transactions.

The simulation is conducted on five file sizes, as well as with and without the security scheme deployed, to measure the communication overheads for uploading and downloading, as shown in Figure 3. We set the file size to vary between 1 MB, 10 MB, 100 MB, 1 GB, and 10 GB, with 256 KB as the transmitted block size. Figure 3 shows that the uploading and downloading time improves with increasing data size. It is mainly caused by factors such as the increase of the block counts and security authentication. Since IPFS uses a decentralized storage mechanism, the larger the file, the more the file blocks will be, and the longer the time required to compute the CIDs. In terms of file uploading time, both EBSS v1 and EBSS v2 schemes are higher than a single IPFS configuration. This is caused by the fact that our EBSS scheme introduces extra communication overhead, involving signing, verification, encryption, and transaction submission. One communication overload introduced by EBSS v1 is 6.816 Kb in size and by EBSS v2 is 8.160 Kb in size. The increased communication overload depends on the number of blocked file. However, these security features introduced have only a small impact. For example, the file uploading time difference, between a single IPFS and EBSS v1, is only roughly 3.0 ms to 12.0 ms, and the difference with EBSS v2 is nearly 6.0 ms to 13.0 ms. In addition, as can be seen in the Figure 3, since the file retrieval only requires a block lookup in the IPFS by CID, the

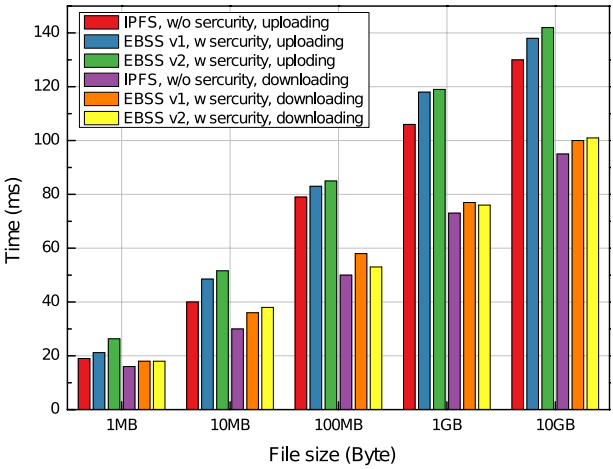


Figure 3 Communication overhead comparison

downloading time for each file is less than the corresponding file uploading time. EBSS access increases the commitment verification overhead only about 2.0 ms, so the file downloading time differs slightly from that of a single IPFS. Therefore, the security properties provided in this paper do not incur expensive communication and transmission overheads.

To measure the storage throughput for EBSS with different numbers of consortium chain nodes, we perform a simulation based on different number of data upload requests. We set the number of nodes to be 4, 8, 12, 16, 20, and 24, and the number of data upload requests to be 100, 500, 1, 000, 2, 000, 5, 000, and 10, 000, respectively. The simulation is conducted based on the EBSS v1 scheme, and the results are shown in Figure 4. Since the communication overhead introduced by the EBSS v2 scheme is slightly higher than that of the EBSS v1 scheme, the throughput of the EBSS v2 scheme is bound to be lower than that of the EBSS v1 scheme. In general, the throughput variation trends of EBSS v2 and EBSS v1 are consistent. As seen from Figure 4, when the number of nodes is 4 and

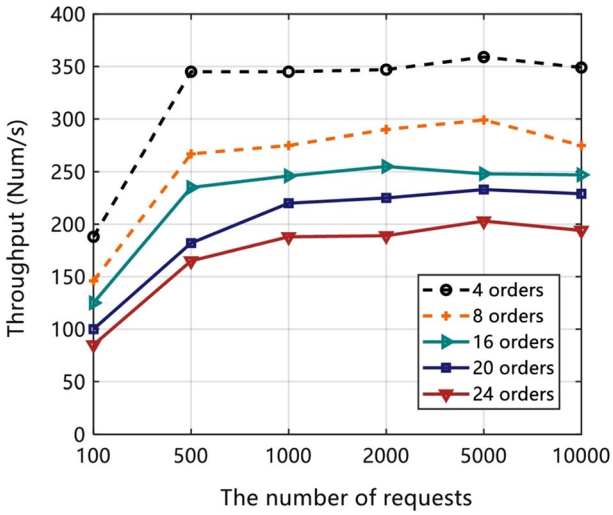


Figure 4 Storage efficiency with different number of nodes

the number of requests is between 100 and 500, the system throughput increases rapidly. In contrast, when the number of requests exceeds 5,000, the throughput fluctuates around 359 transactions per second. This means that the server performance reaches a bottleneck in this interval. For the case of 4 nodes, the upload throughput is up to 359 transactions per second. It should be noted that the transaction processing efficiency of EBSS v1 is negatively correlated to the number of nodes. In case of 24 nodes, the maximum upload efficiency of EBSS v1 is 203 transactions per second, which is 43.45% lower than that of 4 nodes. It is due to the fact that the data written to the blockchain requires all nodes to update the records in the ledger according to the consensus mechanism. As the number of nodes increases, the information exchange between nodes becomes frequent and the time consumption for validating block transactions increases accordingly. Therefore, the more the number of nodes in the consortium chain, the lower the transaction efficiency and throughput will be.

6.4 Comparison with related works

We compare EBSS with four recently proposed schemes [16–19]. All of the schemes focus on satisfying the security services for the blockchain system, including authentication, integrity and provable security. Considering the differences in implementation details and actual deployment, we mainly compare their computational overhead and communication overhead during the authentication process. The detailed computation overhead is compared in Table 3. Schemes [16, 19] use hash-to-point calculation, denoted by H_2, each of them will introduce a computation overhead with 5.493 ms. As shown in Table 3, the computational overhead in EBSS under version 1 is significantly lower than that of the schemes [16, 17, 19], and only slightly higher than that of [18]. In our scheme, anonymous authentication does not require the two parties to interact to calculate and verify the identity information multiple times, but only performs one signature-verification process on the information to complete the authentication. Therefore, the anonymous authentication proposed in this paper is efficient, and it can ensure that the user's identity privacy is not leaked. In fact, the computational overhead of our version 2 is the highest in all schemes. Since version 2 is an enhanced security scheme, it inevitably introduces some extra overhead. In our scheme, M_e and M_m refer to the communication overhead introduced by the commitment process, which can be avoided by pre-computation.

Table 4 shows the detailed computational overheads for these five schemes. It should be noted that an asymmetric ciphertext denoted by $|ct_{asy}|$ is 800 bits; a symmetric ciphertext denoted by $|ct_{sy}|$ is 128 bits; and the user identity λ is 32 bits [19]. The communication overheads shown in schemes [16–19] are 7.552 Kb, 5472 Kb, 7360 Kb and 4480 Kb,

Table 3 Computation overhead comparison

Schemes	Computation overhead	Time (ms)
[16]	$6P + 7\mathbb{G}_1 + 2E_2 + H_2$	53.177
[17]	$3P + 9\mathbb{G}_1$	53.177
[18]	$3P + 7\mathbb{G}_1$	31.436
[19]	$4P + 8\mathbb{G}_1 + 2H_2$	50.014
Ours Version 1	$4P + 11\mathbb{G}_1 + (2M_e + 1M_m)$	45.523 (46.202)
Ours Version 2	$8P + 19\mathbb{G}_1 + (2M_e + 1M_m)$	84.551 (85.230)

Table 4 Communication overhead comparison

Schemes	Communication overhead	Size (Kb)
[16]	$5\|\mathbb{G}_1\| + \|\mathbb{G}_2\| + 2\|\mathbb{Z}_q\| + 2\|\lambda\|$	7.552
[17]	$3\|\mathbb{G}_1\| + 3\|ct_{asy}\|$	5.472
[18]	$6\|\mathbb{G}_1\| + 6\|\mathbb{Z}_q\| + 3\|\lambda\| + 5\|n_T\|$	7.360
[19]	$4\|\mathbb{G}_1\| + 2\|ct_{sy}\| + 2\|\lambda\| + 2\|n_T\|$	4.480
Ours Version 1	$6\|\mathbb{G}_1\| + 4\|\mathbb{Z}_q\| + \|n_T\|$	6.816
Ours Version 2	$7\|\mathbb{G}_1\| + 6\|\mathbb{Z}_q\| + \|n_T\|$	8.160

respectively. It can be seen from Table 4, the communication cost of our scheme under version 1 is significantly lower than the scheme of [16, 18], and only slightly higher than the scheme of [17, 19]. Since version 2 is an enhanced security version, the communication cost of our scheme is the highest among all schemes. Even though the communication overheads are roughly similar for all five schemes, pursuing more security properties, e.g., identity anonymity, perfect hiding, tracking and auditing, hierarchical privilege management, are the key features that distinguish our scheme from others.

7 Further discussions and future work

As mentioned above, EBSS shows some desirable security properties, including confidentiality, anonymity, identity tracking, and so on. For further improvements, there are some issues worth exploring.

First, one possible concern is about reducing the workload on the server, e.g., during the signing and verification, secure transmission, traceability and auditability processes. In our current solution, some specific mechanisms such as anonymous authentication, commitment, and access control are deployed, which inevitably introduce additional server workloads. If considering minimizing these workloads, our proposed scheme can be further optimized. There are two versions of the signature algorithm in this scheme. If only the signature algorithm of EBSS v1 is considered, it will significantly reduce the computational load during the signing and verification process. For example, the calculations of M,N,ν,t and θ in the signature algorithm can be omitted; meanwhile, the calculations of ν', N' and the verification of θ in the verification algorithm also can be omitted. Such optimizations do not change the existing security properties in EBSS. In addition, other possible improvement approach is to use periodic tracking and auditing, or to adopt lightweight hash computation instead of bilinear pairing in this paper.

Second, in this proposed scheme, we do not take the revocation mechanism of transaction records into consideration. The blockchain system will continuously generate new blocks, which means that all transactions will be recorded in the blockchain. Those invalid or useless transaction information will occupy the blockchain for a long time, which may have an impact on the storage performance of the system later. However, it will be our future interest to investigate appropriate security techniques to enable a blockchain solution with revocable transaction records.

Moreover, the proposed authentication approaches in our scheme can effectively defend against most external attackers, but do not investigate mechanisms to resist attacks from internal attackers, e.g., black-hole attacks and Sybil attacks. We assume that the proposed scheme is for trusted and honest internal users. However, if this assumption does not

hold, we should integrate other secure mechanisms to achieve corresponding protection. Although not the focus of this paper, such mechanisms have been well studied in the literature [51, 52], and they are relatively independent from our proposed scheme. However, we believe that effective internal attack detection mechanisms are feasible and will be of interest to us for future work.

8 Conclusion

The secure sharing of financial electronic credentials is a promising research topic. Personal property and private information have been extensively utilized in real estate financial credentials. In this work, we focus on their secure sharing and circulation controllability, and thus propose a blockchain-based secure sharing scheme. Other than the basic security properties such as confidentiality and integrity, our scheme can also provide anonymous authentication, access control, traceability and auditability. We have conducted a comprehensive evaluation to further show the security and feasibility of the proposed scheme. Our future work will focus on fast similarity checking, hierarchical access control, and reduced authentication overhead to enable more efficient blockchain-based secure sharing.

Acknowledgements This work is supported in part by the National Natural Science Foundation of China under Grant 61672029 and Grant 61972366, Open Research Project of The Hubei Key Laboratory of Intelligent Geo-Information Processing.

Author contributions In this work, each author contributed as follows: Yadi Wu is mainly with the research goals evolution, methodological designs, algorithm designs, method validation, and writing and preparation of the original manuscript; Guiyao Tie is mainly with the data collection, software implementation, data and experimental result analysis, and experimental verification, and preparation of the original manuscript; Yao Yu is mainly with the model designs, algorithm validation, formal analysis, experimental results analysis, and writing and preparation of the manuscript; Jianxin Li is in charge of methodological design, formal analysis, algorithm analysis, research activity supervision, critical revision, and writing-review and editing of the manuscript; Jun Song is in charge of research goals evolution, methodological designs, formal analysis, security analysis, research activity supervision and management, critical revision, and in-depth writing-review and editing of the manuscript.

Funding This work is supported in part by the National Natural Science Foundation of China under Grant 61672029 and Grant 61972366, Open Research Project of The Hubei Key Laboratory of Intelligent Geo-Information Processing.

Data availability Our simulation dataset is from "Zillow", which is a public dataset on real estate transactions. Data sets used or analyzed during the current study are available, either by direct request from the following link, i.e., https://www.zillow.com/research/data/, or from the corresponding author on necessary conditions or reasonable request only for academic research purposes.

Code availability Source code used or analyzed during the current study is available from the corresponding author on reasonable request. The source code required to reproduce our work findings is not appropriate to share at this time as the code and data also form part of an ongoing study.

Declarations With regard to this submitted manuscript entitled "EBSS: A secure blockchain-based sharing scheme for real estate financial credentials", we declare as follows:

Ethics approval We declare that the work described in this submission is original research that has not been previously published or under consideration by other conferences or journals.

Consent for publication All authors have agreed to publish this paper due to the original contribution prior to this submission.

Conflict of interests (check journal-specific guidelines for which heading to use): We declare that we do not have any commercial or associative interests that represent a conflict of interest and a conflict of competing interests in relation to the work submitted.

Consent to participate All authors have agreed to participate in this work due to the collaboration and contribution in advance of this submission.

References

1. Tabatabai Hesari, N.: Environmental and human factors of real estate transactions costs and control measures in Iran registration law. J. Encyclopedia Econ. Law **23**(9), 1–20 (2017)
2. Ullah, F., Al-Turjman, F.: A conceptual framework for blockchain smart contract adoption to manage real estate deals in smart cities. Neural Comput. Applic. 1–22 (2021)
3. Jin, B., Song, W., Zhao, K., Li, S., Wang, Z.: Cloud infrastructure and monitoring system for real estate registration. In: 2018 26Th International Conference on Geoinformatics, pp. 1–9 (2018)
4. Chen, Y., Wang, L., Chen, X., Ranjan, R., Zomaya, A.Y., Zhou, Y., Hu, S.: Stochastic workload scheduling for uncoordinated datacenter clouds with multiple qos constraints. IEEE Trans. Cloud Comput. **8**(4), 1284–1295 (2020)
5. Shuaib, M., Alam, S., Daud, S.M.: Improving the authenticity of real estate land transaction data using blockchain-based security scheme. In: Anbar, M., Abdullah, N., Manickam, S. (eds.) Advances in Cyber Security - Second International Conference, ACes 2020, Penang, Malaysia, December 8-9, 2020. Communications in Computer and Information Science, vol. 1347, pp 3–10. Springer, Heidelberg (2020)
6. Usmani, R.S.A., Hashem, I.A.T., Pillai, T.R., Saeed, A., Abdullahi, A.M.: Geographic information system and big spatial data: a review and challenges. Int. J. Enterp. Inf. Syst. **16**(4), 101–145 (2020)
7. Dees, K., Rahman, S.: Enhancing infrastructure security in real estate. arXiv:1512.00064 (2015)
8. Zhang, C., Li, M., Li, Y.: Financial risk analysis of real estate bubble based on machine learning and factor analysis model. J. Intell. Fuzzy Syst. **40**(4), 6493–6504 (2021)
9. Liu, H., Chen, Y.L., Cui, N., Li, J., et al.: An effective data fusion model for detecting the risk of transmission line in smart grid. IEEE Internet of Things Journal (2021)
10. Kalia, P., Bansal, D., Sofat, S.: A hybrid approach for preserving privacy for real estate data. Int. J. Inf. Comput. Secur. **15**(4), 400–410 (2021)
11. Chen, Y., Chen, X., Liu, W., Zhou, Y., Zomaya, A.Y., Ranjan, R., Hu, S.: Stochastic scheduling for variation-aware virtual machine placement in a cloud computing CPS. Future Gener. Comput. Syst. **105**, 779–788 (2020)
12. Ma, X., Wang, C., Chen, X.: Trusted data sharing with flexible access control based on blockchain. Comput. Stand. Interfaces **78**, 103543 (2021)
13. Li, M., Shen, L., Huang, G.Q.: Blockchain-enabled workflow operating system for logistics resources sharing in e-commerce logistics real estate service. Comput. Ind. Eng. **135**, 950–969 (2019)
14. Chen, W., Guo, X., Chen, Z., Zheng, Z., Lu, Y.: Phishing scam detection on ethereum: Towards financial security for blockchain ecosystem. In: Bessiere, C. (ed.) Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020, pp 4506–4512. ijcai.org, Freiburg (2020)
15. Nyaletey, E., Parizi, R.M., Zhang, Q., Choo, K.R.: Blockipfs - Blockchain-enabled interplanetary file system for forensic and trusted data traceability. In: IEEE International Conference on Blockchain, Blockchain 2019, Atlanta, GA, USA, July 14-17, 2019, pp 18–25. IEEE, New York (2019)
16. Hong, H., Hu, B., Sun, Z.: Toward secure and accountable data transmission in narrow band internet of things based on blockchain. Int. J. Distributed Sens. Netw. **15**(4), 1–10 (2019)
17. Harbi, Y., Aliouat, Z., Refoufi, A., Harous, S., Bentaleb, A.: Enhanced authentication and key management scheme for securing data transmission in the internet of things. Ad Hoc Netw. **94**, 101948 (2019)
18. Jia, X., He, D., Kumar, N., Choo, K.R.: Authenticated key agreement scheme for fog-driven iot healthcare system. Wirel. Netw. **25**(8), 4737–4750 (2019)
19. Fan, Q., Chen, J., Deborah, L.J., Luo, M.: A secure and efficient authentication and data sharing scheme for internet of things based on blockchain. J. Syst. Archit. **117**, 102112 (2021)
20. Mani, D., Choo, K.R., Mubarak, S.: Information security in the south australian real estate industry: a study of 40 real estate organisations. Inf. Manag. Comput. Secur. **22**(1), 24–41 (2014)

21. Karamitsos, I., Papadaki, M., Al Barghuthi, N.B., et al: Design of the blockchain smart contract: A use case for real estate. **9**, 177–190 (2018)
22. Choo, K.R., Heravi, A., Mani, D., Mubarak, S.: Employees' intended information security behaviour in real estate organisations: A protection motivation perspective. In: 21St Americas Conference on Information Systems, AMCIS 2015, Puerto Rico, August 13-15, 2015. Association for Information Systems, New York (2015)
23. Liu, X., Huang, H., Xiao, F., Ma, Z.: A blockchain-based trust management with conditional privacy-preserving announcement scheme for vanets. **7**,4101–4112 (2020)
24. Li, Y., Huang, J., Qin, S., Wang, R.: Big data model of security sharing based on blockchain. In: 3Rd International Conference on Big Data Computing and Communications, BIGCOM 2017, Chengdu, China, August 10-11, 2017, pp 117–121. IEEE Computer Society, New York (2017)
25. Shrestha, A.K., Vassileva, J.: Blockchain-based research data sharing framework for incentivizing the data owners. In: Chen, S., Wang, H., Zhang, L. (eds.) Blockchain - ICBC 2018 - First International Conference, Held as Part of the Services Conference Federation, SCF 2018, Seattle, WA, USA, June 25-30, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10974, pp 259–266. Springer, Heidelberg (2018)
26. Gupta, A., Rathod, J., Patel, D., Bothra, J., Shanbhag, S., Bhalerao, T.: Tokenization of real estate using blockchain technology. In: Zhou, J., Conti, M., Ahmed, C.M., Au, M.H., Batina, L., Li, Z. (eds.) Applied Cryptography and Network Security Workshops - ACNS 2020 Satellite Workshops, AIBlock, AIHWS, AIoTS, Cloud S&P, SCI, SecMT, and SiMLA, Rome, Italy, October 19-22, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12418, pp 77–90. Springer, Heidelberg (2020)
27. Zhang, S., Lee, J.: A group signature and authentication scheme for blockchain-based mobile-edge computing. IEEE Internet Things J. **7**(5), 4557–4565 (2020)
28. Chen, L., Ng, S., Wang, G.: Threshold anonymous announcement in vanets. IEEE J. Sel. Areas Commun. **29**(3), 605–615 (2011)
29. Hewa, T.M., Hu, Y., Liyanage, M., Kanhare, S.S., Ylianttila, M.: Survey on blockchain-based smart contracts: Technical aspects and future research. IEEE Access **9**, 87643–87662 (2021)
30. Yaga, D., Mell, P., Roby, N., Scarfone, K.: Blockchain technology overview. arXiv:1906.11078 (2019)
31. Sun, J., Yao, X., Wang, S., Wu, Y.: Non-repudiation storage and access control scheme of insurance data based on blockchain in IPFS. IEEE Access **8**, 155145–155155 (2020)
32. Chaum, D., van Heyst, E.: Group signatures. In: Davies, D.W. (ed.) Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of Of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings. Lecture Notes in Computer Science, vol. 547, pp 257–265. Springer, Heidelberg (1991)
33. Pedersen, T.P.: Non-Interactive and information-theoretic secure verifiable secret sharing. In: Feigenbaum, J. (ed.) Advances in Cryptology - CRYPTO '91, 11Th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1991, Proceedings. Lecture Notes in Computer Science, vol. 576, pp 129–140. Springer, Heidelberg (1991)
34. Zhang, L., Song, J., Pan, J.: A privacy-preserving and secure framework for opportunistic routing in dtns. IEEE Trans. Veh. Technol. **65**(9), 7684–7697 (2016)
35. Chen, Y., Fan, J., Deng, Z., Du, B., Huang, X., Gui, Q.: PR-KELM: Icing level prediction for transmission lines in smart grid. Future Gener. Comput. Syst. **102**, 75–83 (2020)
36. Lysyanskaya, A., Rivest, R.L., Sahai, A., Wolf, S.: Pseudonym systems. In: Heys, H.M., Adams, C.M. (eds.) Selected Areas in Cryptography, 6Th Annual International Workshop, SAC'99, Kingston, Ontario, Canada, August 9-10, 1999, Proceedings. Lecture Notes in Computer Science, vol. 1758, pp 184–199. Springer, Heidelberg (1999)
37. Camenisch, J., Lysyanskaya, A.: Signature schemes and anonymous credentials from bilinear maps. In: Franklin, M.K. (ed.) Advances in Cryptology - CRYPTO 2004, 24Th Annual International Cryptologyconference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings. Lecture Notes in Computer Science, vol. 3152, pp 56–72. Springer, Heidelberg (2004)
38. Chen, L., Morrissey, P., Smart, N.P.: Pairings in trusted computing. In: Galbraith, S.D., Paterson, K.G. (eds.) Pairing-Based Cryptography - Pairing 2008, Second International Conference, Egham, UK, September 1-3, 2008. Proceedings. Lecture Notes in Computer Science, vol. 5209, pp 1–17. Springer, Heidelberg (2008)
39. Rao, F.: On the security of a variant of elgamal encryption scheme. IEEE Trans. Dependable Secur. Comput. **16**(4), 725–728 (2019)
40. Khalili, M., Dakhilalian, M., Susilo, W.: Efficient chameleon hash functions in the enhanced collision resistant model. Inf. Sci. **510**, 155–164 (2020)
41. Berman, I., Degwekar, A., Rothblum, R.D., Vasudevan, P.N.: Multi-collision resistant hash functions and their applications. In: Nielsen, J.B. (ed.) Advances in Cryptology - EUROCRYPT 2018 - 37th

Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II. Lecture Notes in Computer Science, vol. 10821, pp 133–161. Springer, Heidelberg (2020)
42. Metere, R., Dong, C.: Automated cryptographic analysis of the Pedersen commitment scheme. In: Rak, J., Bay, J., Kotenko, I.V., Popyack, L.J., Skormin, V.A., Szczypiorski, K. (eds.) Computer Network Security - 7Th International Conference on Mathematical Methods, Models, and Architectures for Computer Network Security, MMM-ACNS 2017, Warsaw, Poland, August 28-30, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10446, pp 275–287. Springer (2017)
43. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - the Advanced Encryption Standard. Information Security and Cryptography. Springer, Heidelberg (2002)
44. Park, S., Sung, S.H., Chee, S., Yoon, E., Lim, J.: On the Security of Rijndael-Like Structures against Differential and Linear Cryptanalysis. In: Zheng, Y. (ed.) Advances in Cryptology - ASIACRYPT 2002, 8Th International Conference on the Theory and Application of Cryptology and Information Security, Queenstown, New Zealand, December 1-5, 2002, Proceedings. Lecture Notes in Computer Science, vol. 2501, pp 176–191. Springer, Heidelberg (2002)
45. Tsai, K., Leu, F., You, I., Chang, S., Hu, S., Park, H.: Low-power AES data encryption architecture for a lorawan. IEEE Access **7**, 146348–146357 (2019)
46. Lai, R.W.F., Ronge, V., Ruffing, T., Schröder, D., Thyagarajan, S.A.K., Wang, J.: Omniring: Scaling up private payments without trusted setup - formal foundations and constructions of ring confidential transactions with log-size proofs. IACR Cryptol. ePrint Arch. 580 (2019)
47. Demirel, D., Lancrenon, J.: How to securely prolong the computational bindingness of pedersen commitments. IACR Cryptol. ePrint Arch. 584 (2015)
48. Boneh, D.: The Decision Diffie-Hellman Problem. In: Buhler, J. (ed.) Algorithmic Number Theory, Third International Symposium, ANTS-III, Portland, Oregon, USA, June 21-25, 1998, Proceedings. Lecture Notes in Computer Science, vol. 1423, pp 48–63. Springer, Heidelberg (1998)
49. Scott, M., Costigan, N., Abdulwahab, W.: Implementing cryptographic pairings on smartcards. In: Goubin, L., Matsui, M. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2006, 8Th International Workshop, Yokohama, Japan, October 10-13, 2006, Proceedings. Lecture Notes in Computer Science, vol. 4249, pp 134–147. Springer, Heidelberg (2006)
50. Zillow Group: Zillow. https://www.zillow.com/research/data/, Last accessed on 2022–5-1 (2006-2022)
51. Nicolas, K., Wang, Y., Giakos, G.C., Wei, B., Shen, H.: Blockchain system defensive overview for double-spend and selfish mining attacks: a systematic approach. IEEE Access **9**, 3838–3857 (2021)
52. Kumar, P., Kumar, R., Gupta, G.P., Tripathi, R.: A distributed framework for detecting ddos attacks in smart contract-based blockchain-iot systems by leveraging fog computing. Trans. Emerg. Telecommun. Technol 32(6) (2021)