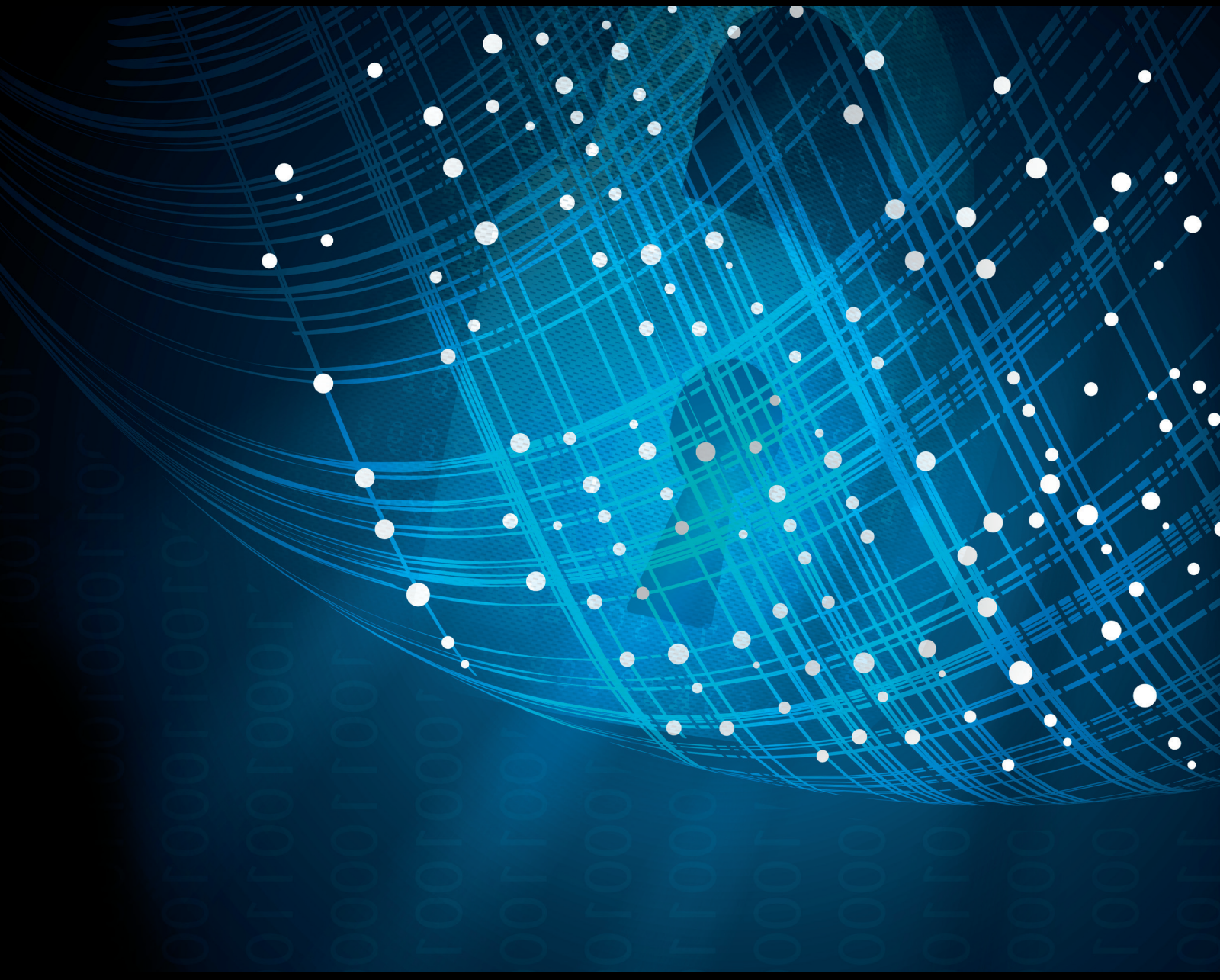
Security and Communication Networks

Theories and Technologies for Securing Blockchain-Empowered Systems

Lead Guest Editor: Willy Susilo
Guest Editors: Robert H. Deng and Yujue Wang



Theories and Technologies for Securing Blockchain-Empowered Systems

Security and Communication Networks

Theories and Technologies for Securing Blockchain-Empowered Systems

Lead Guest Editor: Willy Susilo
Guest Editors: Robert H. Deng and Yujue Wang

Contents

Hindawi
Security and Communication Networks
Volume 2023, Article ID 9559439, 20 pages
https://doi.org/10.1155/2023/9559439

Research Article

Distributed Public Key Certificate-Issuing Infrastructure for Consortium Certificate Authority Using Distributed Ledger Technology

Keita Kumagai,[1] Shohei Kakei,[1] Yoshiaki Shiraishi,[2] and Shoichi Saito[1]

[1]*Nagoya Institute of Technology, Gokiso-cho, Showa-ku, Nagoya, Aichi 466-8555, Japan*
[2]*Kobe University, Rokkodai-cho, Nada-ku, Kobe, Hyogo 657-8501, Japan*

Correspondence should be addressed to Shohei Kakei; kakei.shohei@nitech.ac.jp

Received 12 August 2022; Revised 28 February 2023; Accepted 3 May 2023; Published 7 June 2023

Academic Editor: Yujue Wang

With the development of cloud services and the Internet of Things, the integration of heterogeneous systems is becoming increasingly complex. Identity management is important in the coordination of various systems, and public key infrastructure (PKI) is widely known as an identity management methods. In PKI, a certificate authority (CA) acts as a trust point to guarantee the identity of entities such as users, devices, and services. However, traditional CAs that delegate the operations to a specific organization are not always suitable for heterogeneous services, and a new methodology is required to enable multiple stakeholders to securely and cooperatively operate a CA. In this study, we introduce the concept of a consortium CA and propose a distributed public key certificate-issuing infrastructure that realizes a consortium CA. The proposed infrastructure enables multiple organizations to cooperatively operate a CA suitable for services involving multiple stakeholders. We identify four requirements for the cooperative operation of a consortium CA and design the proposed infrastructure with distributed ledger technology. Furthermore, we present the implementation of smart contracts with Hyperledger Fabric and prove that the proposed infrastructure satisfies the four requirements. Finally, we confirm that certificate issuance and verification are stable at approximately 4 and 3 ms, respectively.

1. Introduction

Identity is critical to security mechanisms such as authentication and access control. Weak identity management mechanisms allow for impersonation attacks. Public key infrastructure (PKI) is widely known as a mechanism for confirming the identity of entities and linking the identity to a public key certificate. In PKI, a certificate authority (CA) acts as a trust point to guarantee the identity of entities, such as users, devices, and services and allows only authenticated entities to connect to the system. A CA issues a public key certificate that contains a public key and the identity of its owner. By verifying the public key certificate, the verifier can believe that the CA guarantees that the owner possesses the public key. A CA is responsible for the authenticity of the content of a public key certificate and is called a single trust point.

CAs can be classified into two types depending on operation forms: public and private. Public CAs are operated as services by socially trusted companies for Internet use. The main role of a public CA is to issue public key certificates to web servers. Public CAs and browser vendors release baseline requirements that all public CAs must follow[1]. Unlike a public CA, which is strictly enforced, private CAs are not required to be as strict as public CAs. A private CA has the same function as a public CA in issuing public key certificates, but a private CA is operated only within a specific domain by a private organization. Thus, a private CA can be used in a form suitable for domain-specific situations: when issuing numerous public key certificates at a high frequency, when issuing public key certificates dedicated to

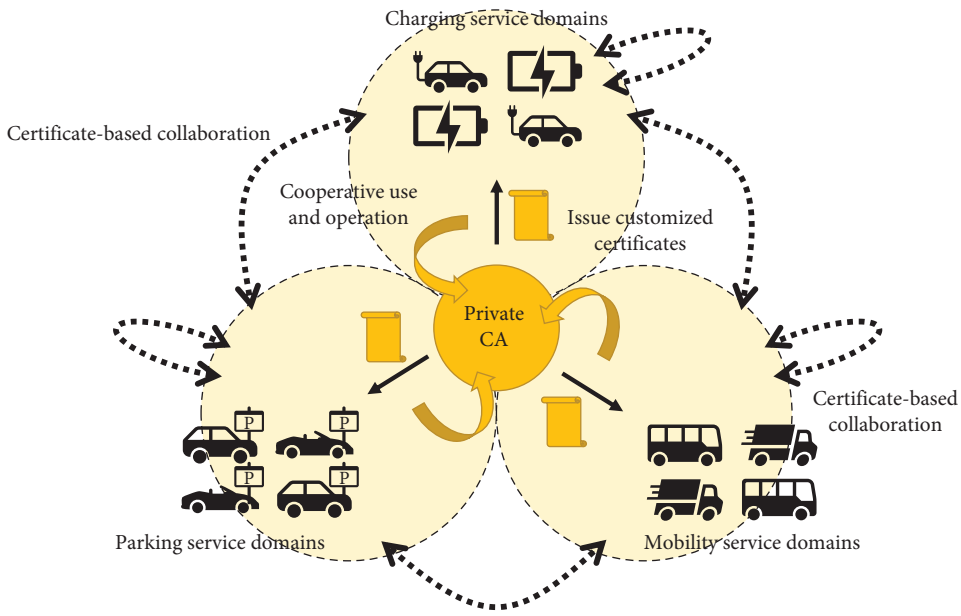


Figure 1: Example of cross-domain services based on [2].

a specific framework, or when policy prohibits the use of a public CA.

With the development of cloud services and wireless technology, various services and devices are being linked together. This is known as the Internet of Things, which involve connecting numerous devices to the Internet, not just people accessing the Internet with a browser. This trend has led to the creation of heterogeneous systems such as cross-domain services [2], cyber-physical systems [3], and digital twins [4]. Such a system involving various stakeholders requires flexible ID management according to individual situations, but it is difficult for public CAs and their additional mechanism, Certificate transparency (CT), to manage IDs flexibly. In CT, issued certificates are recorded in CT logs and made public. According to [5], client certificates published in CT logs can leak private enterprise information, such as business relationships, user growth measurements, and the existence of internal projects prior to their public announcements. The same is true for device certificates, which can reveal what kinds of devices and how many devices company own. In addition, there are financial cost issues with device certificates, and there are cases where public CAs are not suitable. The flexibility of private CAs may be suitable for this issue, but overcoming the stakes and operating a CA securely is challenging. Figure 1 is an example of a cross-domain service [2]. A consumer can use the charging services while using the parking services; this is an interdomain scenario. Moreover, mobility service domains, for instance, may provide route optimization services by integrating and combining different mobility services; this is a single-domain scenario. In this scenario, if a particular organization operates a private CA, the issuance of certificates would be managed by that organization. In such an operation, it is difficult to detect arbitrarily issued certificates. If each organization operates its own private CA, certificate issuance is not dependent on any one organization, but since each organization operates its own CA, the attack points increase. In addition, sufficient security measures must be taken by each organization to ensure that security is breached from the weakest point, that is, a methodology for securely operating a private CA under multiple stakeholders is required. In this study, we propose a distributed public key certificate-issuing infrastructure that allows multiple organizations to cooperatively operate CAs suitable for consortium-type services. In consortium-type services, because multiple organizations cooperate in providing services, the operation of the service should not be influenced by any particular organization. Therefore, to strictly guarantee the identity of service use in consortium-type services, the participating organizations in the consortium must be able to operate CAs in a cooperative manner. However, many security risks are associated with CA operations. The security risks increase if many organizations are involved in the operation of a CA. In this study, we focus on the risk of the unauthorized use of private keys. If a CA's private key is compromised by any organization in the consortium, the organization can issue arbitrary public key certificates using the compromised private key.

For security use of the CA's private key, we employ distributed ledger technology (DLT), also known as blockchain. The proposed infrastructure is built on a DLT system, and all organizations participate in the DLT system. Use of the private key is restricted to smart contracts only in the proposed infrastructure because the availability of the private key in any context leads to arbitrary public key certificates. Smart contracts have business logic agreed upon by all organizations. As long as a private key is used in a smart contract, its use can be considered as authorized by the participating organizations. However, smart contracts do not protect the confidentiality of the data they use; the private key could be compromised during the execution of smart contracts. In the proposed infrastructure, we use

Table 1: X.509 certificate format.

Presigned certificate	Version	
	Serial number	
	Signature algorithm	
	Issuer name	
	Validity period	Not before Not after
	Subject name	
	Subject public key info	Public key algorithm Subject public key
	Issuer unique ID	
	Subject unique ID	
	Extensions	
Signature algorithm		
Signature value		

Hyperledger Fabric Private Chaincode (FPC) [6] and protect the confidentiality of the private key during the execution of smart contracts.

This study contributes to the following:

(i) We introduce the concept of a consortium CA and identify the requirements for operating the consortium CA.

(ii) We design a distributed public key certificate-issuing infrastructure as a consortium CA using DLT.

(iii) We implement a prototype system for the proposed infrastructure with Hyperledger Fabric and evaluate the execution time of smart contracts that perform CA functions. The prototype system maintains the confidentiality of CA's private key using private chaincode, which is an extension of Hyperledger Fabric.

The remainder of this paper is organized as follows: Section 2 describes PKI, Intel Software Guard Extensions (Intel SGX), Hyperledger Fabric, and Hyperledger FPC, which are the technologies used in this proposal. Section 3 describes the concepts of consortium CA and the design policy of this proposal, and Section 4 discusses the data structures and transactions designed to implement the distributed public key certificate-issuing infrastructure. Section 5 describes the implemented smart contract. Section 6 provides a security analysis. Section 7 provides a qualitative and quantitative evaluation. Section 8 compares the proposed infrastructure with those in related studies, and Section 9 concludes the paper.

2. Background

2.1. PKI. PKI is a framework that links a public key and its owner's identity and provides a mechanism for verifying that the communicating party is actually the owner of the public key based on public key cryptography. A CA, which is a single trust point, is responsible for guaranteeing the identity of the owner and issues a public key certificate to the owner after confirming their identity. The CA signs the public key certificate with its own private key to guarantee its contents. The owner can prove its authenticity by presenting the public key certificate to other parties with whom it communicates.

A typical example of a public key certificate is X.509. The X.509 certificate format is standardized by the International Telecommunication Union (ITU). As shown in Table 1, an X.509 public key certificate consists of a presigned certificate, signature algorithm, and signature value. The presigned certificate contains basic information such as a subject name, a validity period, a public key, and extensions.

Public key certificates must be classified according to their purpose, and a major classification is whether they are public key certificates for CAs. Public key certificates for CAs are used to guarantee the ownership of public keys and verify other public key certificates issued by the CA. Whether a certificate is a CA certificate or not is expressed by a dedicated flag in the extensions field of the X.509 certificate.

Figure 2 shows the flow of the application, issuance, and verification processes. In PKI, a certificate holder (CH) presents a public key certificate to a relying party (RP), and the RP identifies the CH with the certificate. Public key certificates are issued by authorities: a registration authority (RA) and a certificate authority (CA). In the application process, the CH generates a key pair and signs a certificate signing request (CSR) with the private key (Step 1-1). The CSR contains all information necessary for the CA to issue a certificate, such as a common name, organization name, and public key of the CH. Then, the CSR is sent to the RA (Steps 1-2). The RA verifies and confirms the identity of the CH (Steps 1–3). After confirming the identity, the application process is completed. In the issuance process, the RA sends the CSR to the CA and requests issuing a public key certificate (Steps 2-1). Upon receiving the request, the CA first creates a presigned certificate based on the CSR. Then, the CA signs the presigned certificate with the CA's private key to create a public key certificate (Step 2-2) and issues this public key certificate to the CH (Steps 2-3). The CA maintains a repository that publishes the issued public key certificates and certificate revocation lists (CRL) and updates the repository as necessary. In the verification process, the CH presents the public key certificate issued by the CA to the relying party to prove the identity of the CH (Steps 3–1). The relying party obtains the CRL and CA's public key certificate (Steps 3-2) and verifies the certificate presented by the CH (Step 3-3). The information retrieved from the repository can be cached; thus, so the relying party does not need to access the repository's every verification process. If the verification is successful, the relying party can trust the identity of the CH.

Figure 3 shows the processes of signing and verifying public key certificates. The signing process is performed by encrypting the hashed presigned certificate with the CA's private key, as shown on the left side of Figure 3, that is, the signature value of the certificate can only be created by the CA. The CA is required to sign the certificate with the agreement of its content, thereby guaranteeing the content of the certificates. Moreover, as shown on the right side of Figure 3, the verification is performed by checking whether

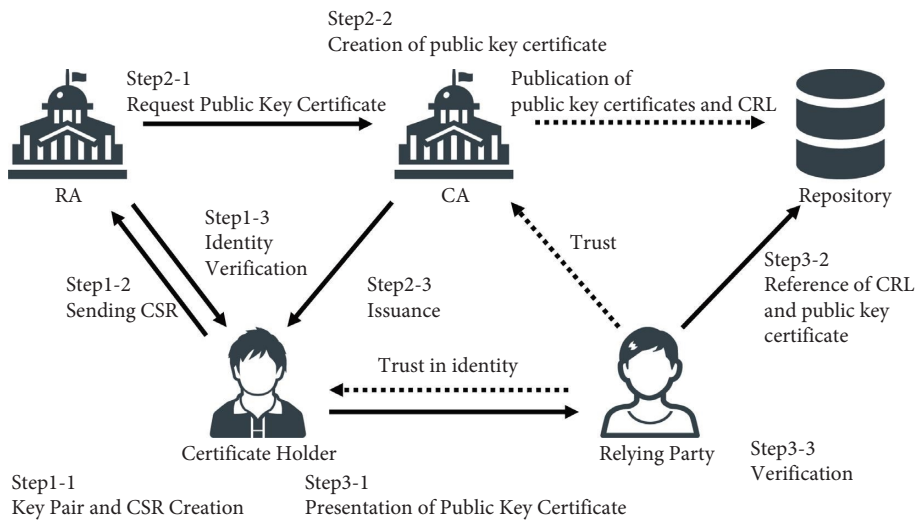


Figure 2: Flow of application, issuance, and verification processes of public key certificates in PKI.

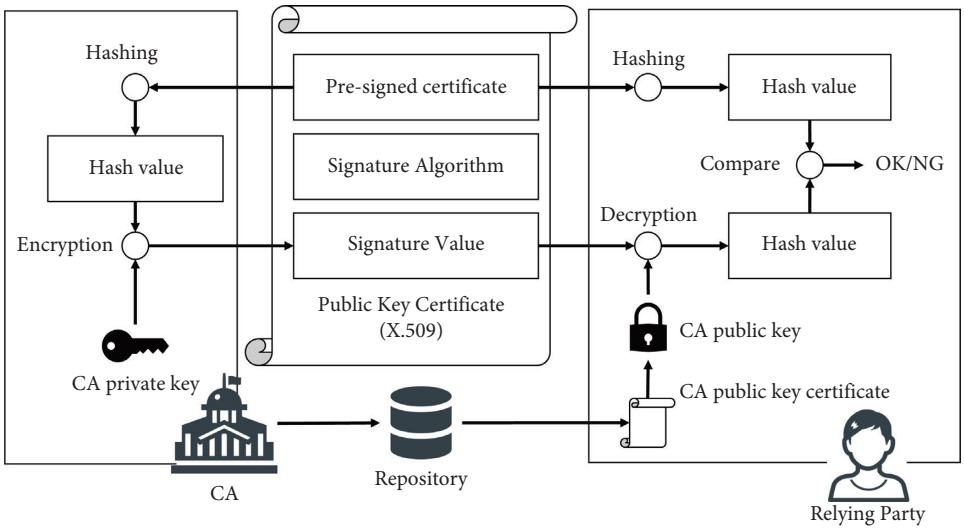


Figure 3: Signing and verification processes for public key certificates.

the hash value of the certificate matches the decrypted signature value with the CA's public key. The CA's public key certificate is publicly available; anyone can verify the certificate issued by the CA. Trust in the contents of a public key certificate is based on the CA properly managing its private key, that is, if a CA's private key is used fraudulently, the contents of a public key certificate can be falsified.

Certificate transparency (CT) [7] and the four cornered trust model [8, 9] are among the latest PKI technologies.

CT is an additional mechanism for increasing certificate transparency, in which the CA stores a record of every certificate it issues in third-party CT logs. By reviewing the CT log, the certificate holder can verify that no fraudulent certificates or certificates from nonpolicy CAs have been issued for his or her domain. This prevents the RP from trusting a fraudulently issued certificate. When a certificate is provided to the CT log, the CT logs return data called a signed certificate timestamp (SCT). The certificate owner presents the SCT with the certificate when it is presented. The RP can use the certificate and SCT to verify that the certificate's data are registered in the CT log.

The four cornered trust model adds an entity called a trust broker to the traditional PKI. The trust broker objectively evaluates the trustworthiness of the CA and its certificates; the RP does not trust the CA directly but trusts this trust broker. The trust broker provides three types of information to the RP:

(i) Quality of Certificate (QoCER): a score from 0 to 1 indicating the level of trust that can be placed in the certificate

(ii) Confidence level (CL): a score from 0 to 1 indicating the degree of confidence the trust broker has in the QoCER recommendation sent to the RP

(iii) Usage information about the recommended or allowed uses of the certificate

Based on the information provided by the trust broker, the RP will determine if the CA is trustworthy. In addition, the trust broker is responsible for the information provided to the RP and must respect and protect the privacy of the RP. The trust broker must be independent of the CA. Possible organizations that operate trust brokers include the following:

(i) A commercial organization whose business is to make recommendations regarding certificates

(ii) Governments wishing to promote electronic commerce in their countries

(iii) An international organization such as the United Nations to facilitate international trade

2.2. Intel Software Guard Extensions. Intel SGX [10, 11] is a trusted execution environment (TEE) that provides a secure execution environment in hardware. Intel SGX creates a cryptographically protected area called an Enclave and executes programs within the Enclave to protect data and execute programs securely. A function called "sealing" encrypts sensitive data within the Enclave, allowing it to be stored outside the Enclave in encrypted form. A function called "unsealing" can then decrypt the encrypted data within the Enclave. Therefore, sensitive data can be protected even outside the Enclave because it is encrypted rather than simply plain text when outside the Enclave.

There is a study [12] that uses Intel SGX to generate keys and certificates. This study seeks to improve the security of key generation and key distribution. The keys are securely generated and encrypted in the secure environment of Intel SGX. The encrypted key is then distributed as an optical signal for secure key distribution.

2.3. Hyperledger Fabric. Hyperledger Fabric [13], which is a DLT framework, is a Hyperledger project [14]. Hyperledger Fabric consists of two types of nodes: peer and orderer. A peer has a distributed ledger, communicates the execution of smart contracts, and updates data in the ledger to the orderer as transactions. The orderer sequences the transactions it receives from the peer and distributes the blocks generated from the transactions to all peers. The Fabric network consists only of authenticated nodes, which join the Fabric network using certificates issued by a dedicated private certification authority. Smart contract installation is based on the permission of each node participating in the Fabric network. Therefore, no particular administrator can arbitrarily decide which smart contracts to install, and only those smart contracts that are agreed upon by all nodes are installed.

Two types of commands are used to invoke smart contracts: invoke, which generates a transaction, and query, which does not generate a transaction. The invoke command can update data in the ledger by generating a transaction, but time is required for all nodes to synchronize their ledgers. Because the query command does not generate a transaction, it can only perform read operations that do not require updating the data in the ledger, thereby eliminating the need for synchronization time.

When a client invokes a smart contract via invoke, the following processes are performed. First, the client requests the creation of a transaction from multiple peers. Upon receiving the request, the peers execute the smart contract and return the result as a transaction to the client. After receiving transactions from all peers, the client sends the transaction to the orderer, which orders the transactions, generates the blocks, and sends the blocks to all peers. Peers verify each transaction in the block, and if no problems are identified, the transaction is reflected in the ledger.

In a DLT, each peer signs a transaction to prove that it has executed it. The key for signing is generated and managed by each peer. In Hyperledger Fabric, each organization can have a CA, and when a node joins the DLT, each CA generates a certificate for that node. When a node joins the DLT, each CA generates a certificate for that node, which is then used to verify the signatures in the blocks. If the keys used for signatures are compromised, there is a possibility of identity theft, so measures to prevent key compromise are considered. Some of the proposed key compromise countermeasures include the use of multisignature [15], the generation of secret keys from biometric information [16], and the use of hardware security module (HSM) [17].

2.4. Private Chaincode. Hyperledger FPC [6] is a mechanism to run Hyperledger Fabric on Intel SGX. The FPC allows and protects the execution of distributed ledgers and smart contracts, which are key-value type databases. The FPC is designed based on [18].

In FPC, the content of a peer can be protected by Intel SGX. Because the smart contract is executed within Enclave, no one can know the details of the smart contract execution and it can be executed securely. Data to be stored in the ledger are encrypted within Enclave and stored in the ledger in an encrypted state. When data are retrieved from the ledger, the data are retrieved in its encrypted state and decrypted in Enclave. At this time, the key for decryption exists only in Enclave. In addition, communication between nodes is protected by secure sockets layer (SSL) communication using a certificate issued at the time of registration.

In a blockchain such as Fabric, a peer executes the smart contract, and the peer can know the execution details of the smart contract. Therefore, utilizing highly private data is unsuitable in blockchains. However, the execution of smart contracts using Intel SGX, such as FPC, can be useful when leveraging highly private data.

A possible use case for FPC is to train a model for detecting brain abnormalities as a convolutional neural network (CNN) [19]. To obtain a highly accurate model, data owned by a single entity are insufficient; more data are required. Therefore, collecting data from many entities is desirable, but regulations under the General Data Protection Regulation (GDPR) and the Health Insurance Portability and Accountability Act of 1996 (HIPAA) make sharing CT scans and MRI images of the brain difficult. In such cases,

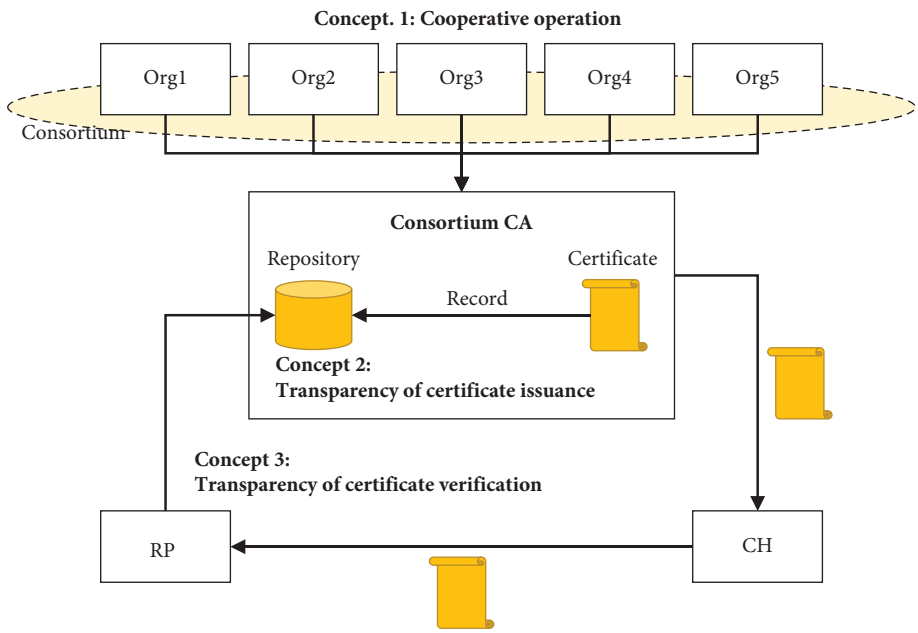


Figure 4: Concepts of consortium CA. This figure shows an example of a consortium consisting of five organizations.

FPC can be used to protect privacy while sharing information.

3. Concepts of Consortium CA and Design Policy

In this section, we introduce the concept of a consortium CA and define the design policies of a distributed public key certificate-issuing infrastructure for a consortium CA. Figure 4 shows the concepts of a consortium CA. Concept 1 is that the consortium CA should be operated by multiple organizations. Organizations participating in the consortium should have equal rights to use the consortium CA and not be restricted in their use by any particular organization. Concept 2 is that the transparency of certificate issuance should be guaranteed. To operate the consortium CA securely, all organizations should enable early detection of unauthorized certificates. Concept 3 is that the transparency of certificate validation should be guaranteed. No organization within the consortium should interfere with the RP's certificate validation.

A naive approach for cooperative operation among multiple organizations is to share the private key of the CA to an administrator of each organization, and issue public key certificates under the responsibility of each administrator. However, this approach is impractical because it requires trusting each administrator, and a malicious administrator can issue unauthorized public key certificates in secret using the private key illegally. For multiple organizations to cooperate in operating a distributed public key certificate-issuing infrastructure, public key certificates must be issued only when necessary at the administrators' discretion while preventing unauthorized issuance of certificates by administrators. Therefore, we define four requirements for the proposed infrastructure.

(1) Req. 1: public key certificates are issued without depending on a specific organization

(2) Req. 2: all public key certificates issued must be recorded

(3) Req. 3: the issuance of a public key certificate cannot be erased

(4) Req. 4: relying parties can confirm that a public key certificate has been issued

Req. 1 comes from Concept 1, Reqs. 2 and 3 come from Concept 2, and Req. 4 comes from Concept 3. Req. 1 prevents an administrator from arbitrarily issuing certificates. This requirement prohibits the delegation of the management of CA's private key to a specific administrator. Req. 2 guarantees the transparency of certificate issuance. This requirement provides accountability around certificate issuance and enables administrators to detect unauthorized certificates. Req. 3 also guarantees the transparency of certificate issuance. This requirement reinforces Req. 2 in terms of the permanence of the record. Req. 4 guarantees the transparency of certificate verification. This requirement allows RPs to check public key certificates without inhibition by any organizations.

We approach these requirements using smart contracts and distributed ledgers with DLT, that is, our approach is to design the logic of certificate issuance with smart contracts and the record of public key certificates with distributed ledgers. Because smart contracts allows transactions to be executed without a third party because the processing logic is predetermined, a public key certificate can be issued based on the logic agreed upon by all participants. This feature satisfies Req. 1. Although the logic may be shared, public key certificates may be issued independently of smart contracts if the private key can be used outside of smart contracts. In this case, such certificates are not recorded in distributed ledgers.

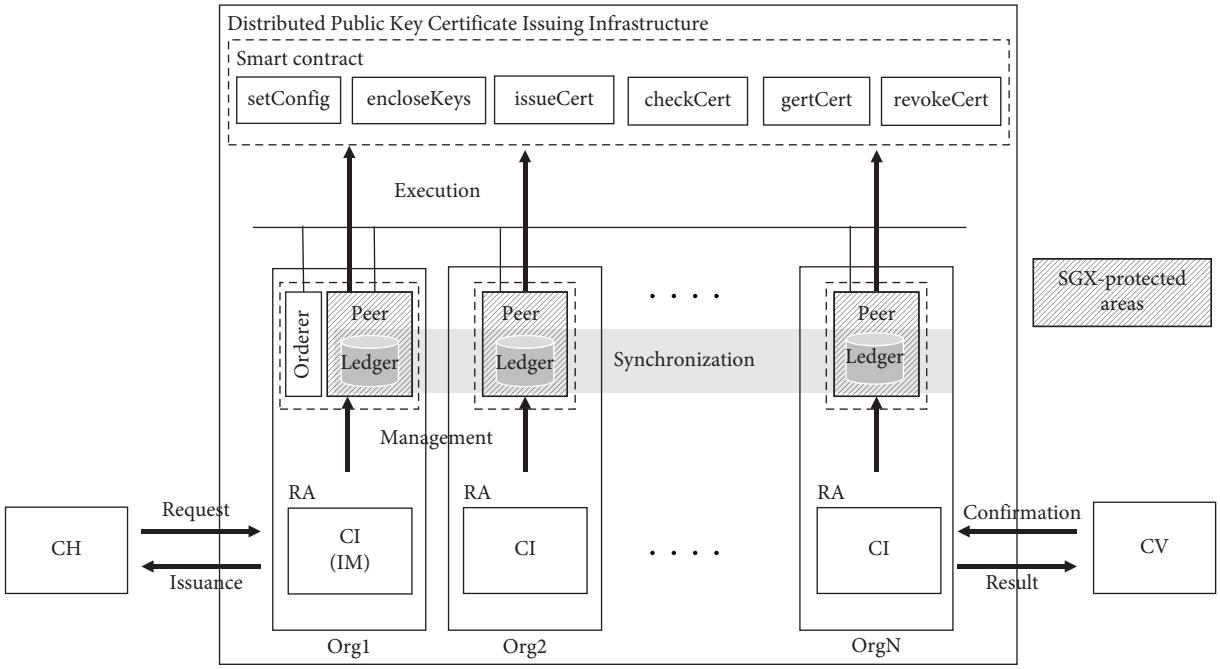


Figure 5: Structure of distributed public key certificate-issuing infrastructure.

Therefore, we limit the use of the private key within smart contracts. In this study, we refer making a specific private key used to sign public key certificates available only for a specific smart contract as "enclosing private keys." We approach Req. 2 with this idea. From Reqs. 1 and 2, distributed ledgers ensure that a record of issued public key certificates is maintained in a manner that cannot be tampered. Consequently, this is expected to conform with Req. 3.

Based on the design, public key certificates can be managed in distributed ledgers and placed under the control of the smart contract. The transparency of the issuance and management of public key certificates in the consortium CA is guaranteed. From this approach, we determine how to employ permissioned DLT as the proposed infrastructure. There are two reasons for this determination. First, designing an infrastructure that can be operated only among the identified organizations is possible. Because the CA functions are implemented using smart contracts, limiting entities that can use the functions is necessary. Permissionless DLT allows an unspecified number of entities to use smart contract. Second, permissionless DLT does not provide transaction finality. The lack of finality causes a potential risk of revoking an issued public key certificate. In permissioned DLT, transactions can be finalized by agreement among the participants. For these reasons, realizing the proposed infrastructure is desirable using permissioned DLT.

Finally, from the decision to use permissioned DLT, Req. 4 is derived. The decision to use permissioned DLT reduces the transparency of certificate verification. Because DLT system participants can only execute smart contracts, verification transparency is also required against a relying party, which is not a participant in the DLT system. This study incorporates the idea that the RP directly verifies the certificate verification results produced by the smart contract. This idea allows RPs to verify certificates without interfering with the organizations.

In this study, we design a distributed public key certificate-issuing infrastructure that satisfies the four requirements using Hyperledger Fabric, which is a permissioned DLT framework. Furthermore, we enclose private keys using FPC. The CA's private keys are securely used in smart contracts' Enclave.

4. Design of a Distributed Public Key Certificate-Issuing Infrastructure

4.1. Overview. Figure 5 shows the structure of the proposed infrastructure. The functionality of a CA is implemented by smart contracts on the FPC network and multiple organizations corresponding to RAs manage peers, which form the FPC network. Each RA can use CA functions by executing smart contracts through peers. In the FPC network, the ledgers held by each peer are synchronized, and the same information is written in all ledgers. Each organization runs a peer on an Intel SGX-enabled device. This ensures that the processing content and input/output of the smart contracts executed by peers, as well as the input/output to the ledgers, are maintained secretly.

The entities that comprise the distributed public key certificate-issuing infrastructure are as follows.

(1) Certificate holder (CH): A certificate holder is an entity that requests a certificate issuance to the proposed infrastructure.

(2) Certificate verifier (CV): A certificate verifier is an entity that verifies certificates.

Table 2: Data stored in the distributed ledger.

Key	Value
"CA." + hash (pk_{CA})	CA private key sk_{CA} Serial number of $cert_{CA}$ Private key sk_{Audit} Serial number of $cert_{Audit}$
"validKey"	"CA." + hash (pk_{CA})
"SerialNumber"	Serial number counter value
Serial number of a public key certificate	Public key certificate
"CRL"	List of a serial number of revoked public key certificate, revocation date, and reason code
"Config"	Issuer information Maximum expiry date

(3) Certificate issuer (CI): A certificate issuer is an entity that invokes a smart contract to issue a certificate.

(4) Infrastructure manager (IM): An infrastructure manager is the entity that sets up the proposed infrastructure. It is responsible for setting up the FPC network, deploying smart contracts, and creating and enclosing key pairs used to create public key certificates. After setup, it acts as a certificate issuer.

(5) Smart contract (SC): A smart contract is a program that reads and writes data on a distributed ledger in DLT. In the distributed public key certificate-issuing infrastructure, it realizes the function of a CA.

(6) Data store (DS): A data store is a ledger for storing data. In a distributed public key certificate-issuing infrastructure, the ledger is protected by Intel SGX and stores the private keys used for issuance and public key certificates issued.

4.2. Data Structure. The distributed ledger stores the CA's private key, public key certificates issued by the distributed public key certificate-issuing infrastructure, and revoked public key certificate information.

Table 2 presents the structure of the data stored in the distributed ledger.

The distributed ledger holds two types of private keys sk_{CA} and sk_{Audit}. The former is a private key for signing public key certificates, and the latter is a private key for guaranteeing verification results. In addition, a public key certificate corresponding to each private key is also stored. These data are stored using pointers, which is the string concatenated string "CA" and hash values of a public key pk_{CA} corresponding to sk_{CA}, as a key.

Because the hash value of pk_{CA} is unique, a different key is generated for each key encapsulation, allowing multiple private keys to be stored. To identify which of the multiple private keys will be used to sign public key certificates, the pointer of the valid private key is stored with string "validKey" as a key.

Then, the counter value of the serial number assigned to the public key certificate is stored with string "SerialNumber" as a key, and the issued public key certificate is stored with its serial number as a key.

Revoked public key certificates are stored with the serial number, revocation date and time, and reason codes of the revoked public key certificate as the key "CRL." Finally, the issuer's information and maximum validity period of the public key certificate are stored with string "Config" as the key.

4.3. Transaction. The distributed public key certificate-issuing infrastructure has four transactions: setup, certificate issuance, certificate verification, and certificate revocation. The variables of the proposed infrastructure are listed in Table 3.

4.3.1. Setup. Figure 6 shows the sequence of the setup. The setup prepares for the issuance of public key certificates in the distributed public key certificate-issuing infrastructure after the FPC network is activated. First, the necessary information for certificate issuance, such as issuer information and maximum validity period, is set by executing setConfig (Step 1). In setConfig, the issuer information and maximum expiration date are stored (Steps 2-3). Then, two key pairs are generated. sk_{CA} is used for certificate issuance, and sk_{Audit} is used during the certificate verification process (Step 4). Then, the generated key pair is enclosed in a distributed ledger by executing encloseKeys (Steps 5–7). Finally, the IM deletes the generated keys (Step 8).

4.3.2. Certificate Issuance. Figure 7 shows the sequence of certificate issuance. In a certificate issuance transaction, a public key certificate is issued to a CH. First, a CH creates a CSR and sends it to a CI, which requests for certificate issuance (Steps 1-2). Then, the CI executes issueCert to issue the certificate and returns it to the CH (Steps 3–6).

4.3.3. Certificate Verification. Figure 8 shows the sequence of certificate verification. In a certificate verification transaction, a CV checks the validity of the CH's certificate $cert_{CH}^{'}$. First, the CV sends $cert_{CH}^{'}$ and a random number $r1$ to a CI to check whether the public key certificate $cert_{CH}^{'}$ to be verified is registered in the distributed public key certificate-issuing infrastructure (Step 1). The CI executes checkCert with $cert_{CH}^{'}$ and the random number $r1$ as

Table 3: List of variables used in this paper.

Variable name	Description
sn	Serial number that uniquely identifies a public key certificate
iss	Issuer name of the public key certificate, which represents the distributed public key certificate-issuing infrastructure
sub	Subject name of the public key certificate
med	The maximum expiration date of public key certificates that can be issued by the distributed public key certificate-issuing infrastructure
rd	The date when a public key certificate issued by the distributed public key certificate-issuing infrastructure is revoked
rc	Reason code that indicates the reason why a public key certificate issued by the distributed public key certificate-issuing infrastructure has been revoked
ed	The expiration date of the public key certificate
csr	This is the certificate signing request to issue the subject's public key certificate

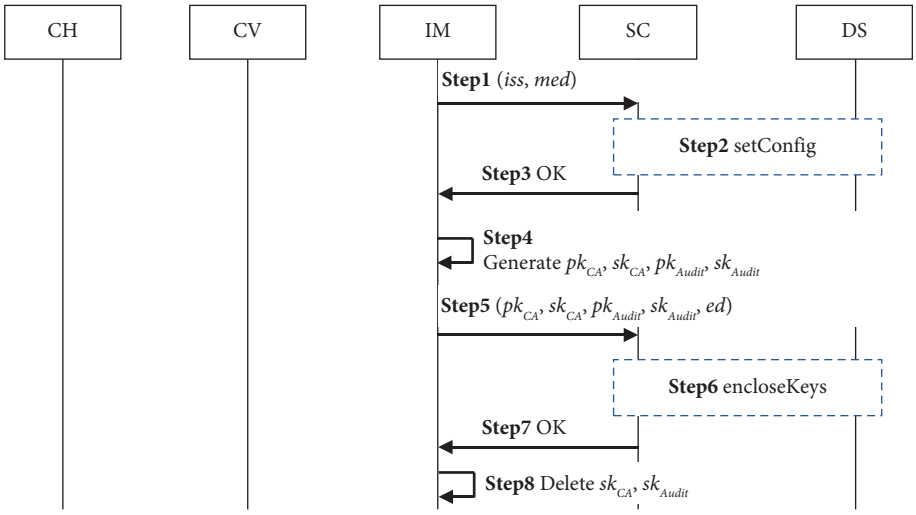


Figure 6: Setup sequence.

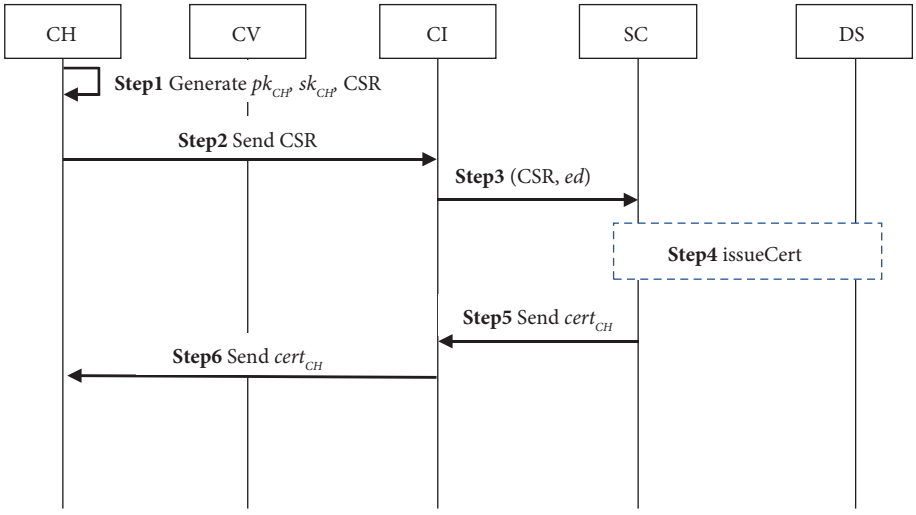


Figure 7: Certificate issuance sequence.

arguments and checks whether cert_{CH}' is a certificate stored in the ledger (Steps 2-3). If it can be confirmed, the SC signs $r1$ with sk_{Audit} and returns $\text{cert}_{\text{Audit}}$ and sig_{r1} (Steps 3-4), and the CI sends them to the CV (Step 5). If not stored in the ledger, cert_{CH}' is not a public key certificate issued by the distributed public key certificate-issuing infrastructure, and the CV is notified of this. Finally, the CV performs the verification (Step 6). In verification, after verifying $\text{cert}_{\text{Audit}}$

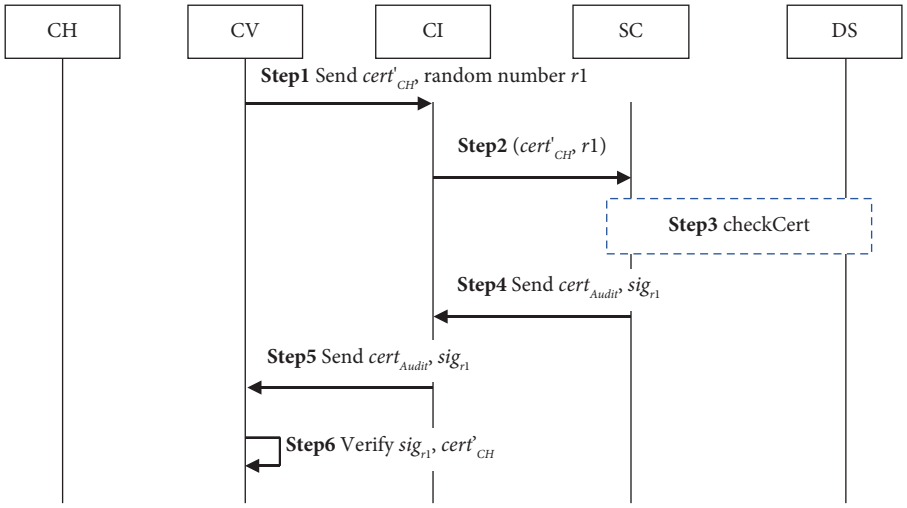


Figure 8: Certificate verification sequence.

using cert_{CA}, sig_{r1} can be verified using $\text{cert}_{\text{Audit}}$ to confirm that it has been processed by the SC execution. Then, $\text{cert}_{CH}^{'}$ is verified using cert_{CA}. It is noted that cert_{CA} is assumed to be known in advance.

4.3.4. Certificate Revocation. Figure 9 shows the sequence of certificate revocation. The certificate revocation transaction revokes a public key certificate. First, a CH makes a revocation request and sends the serial number of the public key certificate to be revoked to a CI (Step 1). The CI executes getCert and retrieves the public key certificate with its serial number as a key (Steps 2–4). Then, identification is required to confirm that the retrieved public key certificate belongs to the CH. To verify the identity, the CI requests the CH to sign a random number $r2$. The CI sends a random number $r2$ to the CH, and the CH generates a signature sig_{r2} for $r2$ using their private key (Steps 5–8). The CI verifies sig_{r2} with cert_{CH} (Step 9). If sig_{r2} is verified, the public key in cert_{CH} corresponds to CH's private key, which confirms that cert_{CH} belongs to the CH. Then, the public key certificate is revoked by executing revokeCert with the serial number of cert_{CH}, revocation date and time, and reason code as arguments (Steps 10–13).

5. Smart Contract Implementation

Six SCs are implemented, and the FPC's SCs can be developed using C++. The OpenSSL library is used to create and sign public key certificates. Table 4 provides the functions used in the proposed infrastructure.

5.1. setConfig. Algorithm 1 provides the pseudocode of setConfig. In setConfig, the putState function stores the issuer information and the maximum validity period with string "Config" as a key.

5.2. encloseKeys. Algorithm 2 provides the pseudocode of encloseKeys. encloseKeys takes key information and an expiration date as arguments. First, in line 16, the serial number and configuration information are obtained by the getState function to obtain the information necessary to generate a public key certificate. Two public key certificates are generated: cert_{CA} with pk_{CA} as the public key and $\text{cert}_{\text{Audit}}$ with pk_{Audit} as the public key. In line 3, a public key certificate is generated, and the issuer information, subject information, serial number, and expiration date are set as the certificate information. Then, the public key certificate is signed with sk_{CA}. Then, in line 22, a hash value of pk_{CA} is calculated, and the key information is stored in a ledger with string "CA.hash (pk_{CA})" as a key. String "CA.hash (pk_{CA})" is also stored with string "validKey" as a key, and the serial number is incremented.

5.3. issueCert. Algorithm 3 provides the pseudocode of issueCert. issueCert takes the CSR and expiration date as arguments. First, to obtain the information necessary to generate a public key certificate, the serial number, configuration information, and string "CA.hash (pk_{CA})" are obtained using the getState function, and key information is obtained using string "CA.hash (pk_{CA})" as a key. Then, from line 5, a public key certificate is generated. The certificate version is set to "3" to use the extended part of the public key certificate. In addition to setting the issuer information, subject information, serial number, and expiration date as certificate information, string "CA.hash (pk_{CA})" is inserted into the Authority Key Identifier area of the extended part of the public key certificate. This allows for the identification of which key signed the public key certificate during verification. In line 16, the public key certificate can then be signed with sk_{CA}, and a public key certificate can be generated. Finally, the generated public key certificate is stored in the ledger, and the serial number is incremented using the putState function.

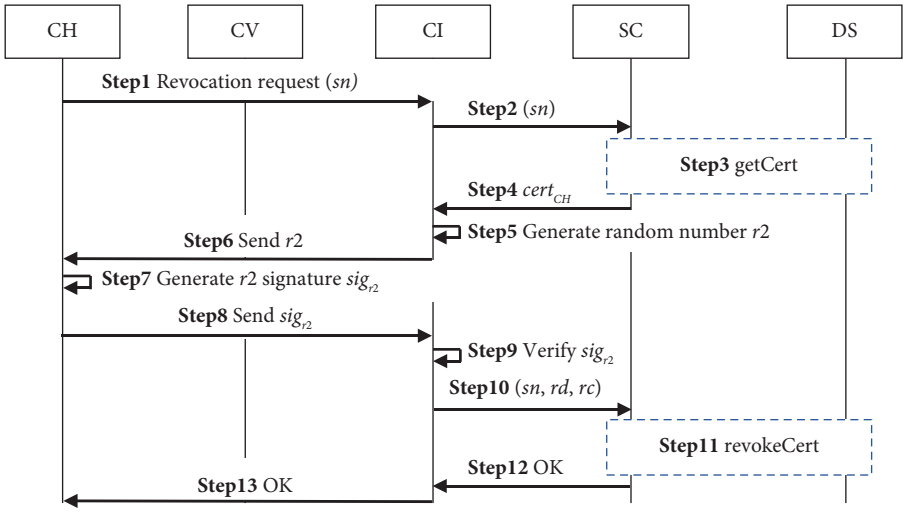


Figure 9: Certificate revocation sequence.

Table 4: List of functions used in this paper.

Function name	Description
putState (k, v)	Store in the ledger as key: k, value: v
$v \leftarrow$ getState (k)	Retrieve the value: v corresponding to key: k from the ledger
signCert (cert, sk)	Sign cert using sk

Input: *iss*, *med*
Output: *status*
(1) putState ("Config," (*iss*, *med*))
(2) **return** "OK"

Algorithm 1: setConfig.

5.4. checkCert. Algorithm 4provides the pseudocode of checkCert. checkCert first retrieves the serial number from the public key certificate $\text{cert}_{CH}^{'}$ using the getState function. From the serial number, it retrieves the public key certificate cert stored in the ledger and compares $\text{cert}_{CH}^{'}$ with the cert. This comparison determines whether $\text{cert}_{CH}^{'}$ is a public key certificate stored in the ledger. In line 6, the getState function obtains the CRL and checks the revocation status of $\text{cert}_{CH}^{'}$. Then, the certificate is signed to guarantee that it has been processed by SC execution. In line 15, to identify the key that issued $\text{cert}_{CH}^{'}$, the extension from $\text{cert}_{CH}^{'}$ is extracted. The extension contains CA.hash (pk_{CA}), which can be used as a key to retrieve the key information. After extracting the key information, sign $r1$ with sk_{Audit} and generate sig_{r1}. Then, $\text{cert}_{\text{Audit}}$, which is necessary for verifying sig_{r1}, is extracted and output.

5.5. getCert. Algorithm 5 provides the pseudocode of getCert. In getCert, the public key certificate is retrieved from the serial number using the getState function.

5.6. revokeCert. Algorithm 6 provides the pseudocode of revokeCert. In revokeCert, the CRL stored in the ledger is retrieved. Then, *sn*, *rd*, and *rc* are added to the CRL, and the updated CRL is stored with string "CRL" as a key.

6. Security Analysis

In Section 6.1, we evaluate whether the proposed infrastructure satisfies the four requirements indicated in the design policy. In Section 6.2, we establish the threat model and we perform the security analysis based on the threat model in Section 6.3.

6.1. Evaluation of Meeting Requirements

6.1.1. Evaluation for Req. 1. Because a public key certificate is signed with a private key to guarantee the issuing entity, a CA is required to strictly manage the private key. For multiple organizations to cooperate in operating the proposed infrastructure, CIs must be able to use the CA's private key, but simply sharing the private key increases the risk of private key leaks and unauthorized use.

```
Input: sk_CA, sk_Audit, pk_CA, pk_Audit, ed
Output: status
function genCert (sk, pk, conf, cnt, ed)
  //Set issuer information, subject information, serial number and public key
  cert.iss ⟵ conf.iss
  cert.sub ⟵ conf.iss
  cert.sn ⟵ cnt
  cert.pk ⟵ pk
  if ed does not exceed med then
    cert.ed ⟵ ed//Set expiration date
  else
    return Error
  end if
  signCert (cert,sk)//Sign a certificate
  return cert
end function
cnt ⟵ getState ("serialNumber")
conf ⟵ getState ("Config")
cert_CA ⟵ genCert (sk_CA, pk_CA, conf,cnt, ed)
cert_Audit ⟵ genCert (sk_CA, pk_CA, conf,cnt + 1, ed)
putState (cert_CA.sn, cert_CA)
putState (cert_Audit.sn, cert_Audit)
hash ⟵ Hash (pk_CA)
putState ("CA." + hash, sk_CA, sk_Audit, cert_CA.sn, cert_Audit.sn)
putState ("validKey," "CA." + hash)
putState ("serialNumber," cnt + 1)
return "OK"
```

Algorithm 2: encloseKeys.

```
Input: csr, ed
Output: cert
cnt ← getState ("serialNumber")
conf ← getState ("Config")
validkey ← getState ("validKey")
(sk_CA, sk_Audit, CA.sn, Audit.sn) ← getState (validkey)
cert.version ← 3
cert.iss ← conf.iss
cert.sub ← csr.sub
cert.sn ← cnt
cert.pk ← csr.pk
cert.extension ← validkey
if ed does not exceed med then
  cert.ed ← ed
else
  return ← Error
end if
signCert (cert, sk_CA)
putState (cert.sn, cert)
putState ("serialNumber", cnt + 1)
return cert
```

Algorithm 3: issueCert.

In the proposed infrastructure, a series of certificate creation processes, including the signing process using the private key, is implemented as an SC to avoid dependence on a specific CI, as shown in Algorithm 3. Since the private keys and the SC execution are protected by Intel SGX, all the CIs have fair access to the private keys and cannot interfere with other CIs. In this manner, the proposed infrastructure satisfies Req. 1.

```
Input: cert'_CH, r1
Output: sig, cert_Audit
//Retrieve a certificate using cert'_CH serial number
cert ← getState (cert'_CH.sn)
ifcert and cert'_CH are not matched then
    return Error ("Cert is invalid")
end if
crl ← getState ("crl")
i ← 0
//Confirm revocation status
whilecert.sn ! = crl[i].sn do
    i ← i + 1
end while
ifcert is revoked then
    return Error ("Cert is revoked")
end if
extension ← cert'_CH.extension
(sk_CA, sk_Audit, CA.sn, Audit.sn) ← getState (extension)
sig ← sign(r1, sk_Audit)//Sign r1 using sk_Audit
cert_Audit ← getState (Audit.sn)
returnsig, cert_Audit
```

Algorithm 4: checkCert.

```
Input: sn
Output: cert
cert ← getState (sn)
return cert
```

Algorithm 5: getCert.

```
Input: sn, rd, rc
Output:status
crl ← getState ("crl")//Retrieve crl
crl ← add (sn, rd, rc)//Add revocation information to crl
putState ("CRL," crl)
return "OK"
```

Algorithm 6: revokeCert.

6.1.2. Evaluation for Req. 2. Each organization cooperates to maintain the proposed infrastructure; ans simultaneously plays a role in monitoring the other organizations to ensure that no fraudulent public key certificates are issued. Because the public key certificate creation process is defined by the SC, a certain level of security is guaranteed, such as prohibiting the use of weak cryptographic algorithms. However, being able to evaluate items that cannot be evaluated uniformly, such as the eligibility of the subject of issuance and validity of the expiration date, using public key certificates that have actually been issued is desirable.

In the proposed infrastructure, the SC must be performed to issue a public key certificate. The certificate issuance process is realized as a single transaction of issuance and recording, with public key certificates being stored in the distributed ledger at the same time they are created. This ensures that all public key certificates issued are recorded in the distributed ledgers. By managing the public key certificates in the distributed ledgers, the public key certificates issued are shared by all organizations. Meanwhile, the proposed infrastructure requires the infrastructure manager to enclose the private keys in the distributed ledgers. The IM must confirm that the enclosed private keys have been securely deleted, as shown in Figure 6. If the private keys are not deleted, the IM can create public key certificates in secret. However, public key certificates created without executing an SC are not recorded in the distributed ledgers; thus, only public key certificates that have been legitimately issued are recorded. The certificate verification sequence also checks whether a public key certificate is registered in the

distributed ledgers. Because public key certificates created in secret are not recorded in the distributed ledger, they can be detected by the certificate validation sequence. In this manner, the proposed infrastructure satisfies Req. 2.

6.1.3. Evaluation for Req. 3. If the issuance of a public key certificate can be erased, inconvenient issuance facts can be erased later. Because the issuance of a public key certificate is recorded as the storage of the public key certificate in distributed ledgers, the erasure of the issuance of a public key certificate means the erasure of the public key certificate stored in the distributed ledgers.

In relation to Req. 2, even if the public key certificates issued are recorded in the distributed ledgers without omission, fraudulent public key certificates cannot be detected if the records are being tampered with or deleted. The public key certificates stored in a distributed ledger cannot be deleted based on the distributed ledger's tamper resistance, which is generally provided by DLT. In addition, the proposed infrastructure, no SC can delete public key certificates recorded in the distributed ledgers, and the public key certificates cannot be deleted by abusing legitimate SCs. Even if SCs cannot delete public key certificates, issuance of public key certificates cannot be confirmed if they are no longer recognized by the proposed infrastructure. Therefore, public key certificates issued are stored with their hash value as a key to ensure that they are stored uniquely without being overwritten. In this manner, the proposed infrastructure satisfies Req. 3.

6.1.4. Evaluation for Req. 4. The verification of public key certificates involves checking that the public key certificate is signed with sk_{CA} and that the public key certificate is recorded in the distributed ledgers. As shown in Figure 5, the existence of this record is verified by a CI who has the authority to execute the SC. There is a risk concerning the CI returning incorrect results because the CI can replace the verification results.

The proposed infrastructure uses sk_{Audit} to sign the verification results in addition to $\mathrm{sk}_{\mathrm{CA}}$ to sign public key certificates. The SC that performs the verification process signs the verification results with sk_{Audit} enclosed in the distributed ledgers. Therefore, the verification results are guaranteed to originate from the SC. The CV can verify the existence of public key certificates without unauthorized intervention by the CI. In this manner, the proposed infrastructure satisfies Req. 4.

6.2. Threat Models. Since the proposed infrastructure allows CIs to issue certificates, we conduct threat modeling by assuming a CI to be a malicious entity. The attacker's goal is to create a fraudulent certificate that will pass authentication checks. From the modeling assumption, the attacker has the capability of a CI. Therefore, the attacker is able to propose deploying smart contracts, agree upon processing of smart contracts, and execute smart contracts. According to the attacker's capability, there can be two types of attacks: deploying a smart contract with incorrect process injected and intervening in the verification process. In the former type, the attacker proposes a smart contract including the function that is advantageous in attacking. In this paper, we analyze the possibility of a smart contract that can issue a CA certificate. In the latter type, the attacker illegally intervenes in the verification process to avoid detection of invalid certificates. If the attacker obtains sk_{CA}, the attacker can generate invalid certificates using it. In this type of attack, the attacker needs to spoof that the invalid certificates are recorded on the distributed ledger.

6.3. Security Analysis Based on Threat Models. For the attack of issuing a CA certificate, if the attacker can generate a CA certificate by issueCert, the attacker uses a private key of that CA certificate and can issue certificates with that CA certificate as their parent. In this case, the CA certificate issued by the attacker is an intermediate certificate for the root CA certificate stored in the distributed ledger. This attack is detectable from two points of view: installation of smart contracts at each CI and verification of certificates. For the first point, the attacker proposes a smart contract, which includes the function issuing a CA certificate to all CIs, and all CIs have to install that smart contract. Since CA certificates are also a type of public key certificate [20], they can essentially be generated in the proposed infrastructure. However, the difference can be determined by the usage of certificates expressed in the extension area of certificates (e.g., Basic Constraints and Key Usage). Each CI is required to understand these differences and decide whether or not to install smart contracts. Although the specific scheme is the future work of this study, it is necessary to establish a policy to determine whether or not to install smart contracts and a system that allows all CIs to check the policy mechanically. For the second point, although the intermediate certificate generated by the attacker is recorded in the distributed ledger, subordinate certificates of that intermediate certificate are not recorded in the distributed ledger. Therefore, invalid certificates can be detected by checkCert.

For the attack of spoofing certificate issuance records on the distributed ledger, the attacker attempts to spoof that certificates are recorded in the distributed ledger on the certificate verification sequence. Since this attack falsifies certificates created without using issueCert as legitimate certificates, it is expected to be used in conjunction with the attack of issuing an intermediate CA certificate. In this attack, the attacker attempts to manipulate the checkCert results either by altering the checkCert results or by falsifying the records of the distributed ledger. The checkCert results are guaranteed by Req. 4, as shown in Section 6.1.4. In addition, it is not possible to spoof certificate issuance records in the distributed ledger since the attacker cannot inject invalid records into the distributed ledger by Req. 2 and Req. 3. If the attacker can obtain sk_{Audit}, the attacker can generate the proof without checkCert. However, the attackable period is very short since sk_{Audit} is deleted by the IM at Step 8 in the setup sequence.

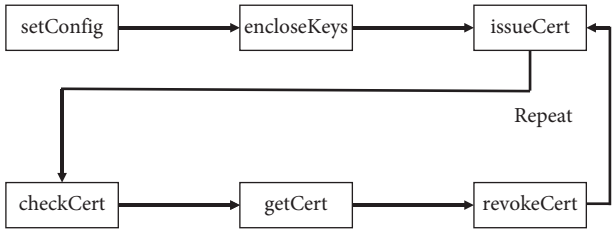


Figure 10: Execution environment.

7. Experimental Evaluation

7.1. Experiment's Summary. We evaluated the basic performance of the proposed infrastructure. Our evaluation aims to analyze the proposed infrastructure from the following three perspectives:

(i) The trend in the time required for the certificate issuance

(ii) The trend in the time required for the certificate verification

(iii) The trend in the time required for the certificate revocation

In the experiments, we executed the proposed SCs in the order shown in Figure 10. After setting up the proposed infrastructure with setConfig and encloseKeys, we repeatedly issued, verified, and revoked certificates 100 times with issueCert, checkCert, getCert, and revokeCert. Within each trial, we measured the execution time for each SC.

To conduct the experimental evaluation, we implemented the system shown in Figure 11. Two peers and one orderer were created as containers, and an FPC network was constructed with them. Our proposed SCs were executed on the FPC network, and its processing time was measured. The measurement was performed with SGX in simulation mode.

7.2. Evaluation Results. The execution times are shown in Table 5, and time to update the status of all ledgers is shown in Table 6. IssueCert and revokeCert write data to the ledger. Therefore, an operation to update the status of all ledgers is necessary to synchronize the ledgers. Looking at the first execution time, setConfig, getCert, and revokeCert required less than 1 ms, whereas encloseKeys, issueCert, and checkCert required several ms. setConfig and getCert have short execution time due to their small amount of processing.

First, we describe the evaluation results of the certificate issuance where issueCert is used. From Figure 12, issueCert has the longest execution time, and it is within a range of approximately 3.9 to 4.2 ms. As shown in Algorithm 3, since issueCert only uses the input data to create a certificate, the execution time is considered to be constant and consistent with the evaluation results. From Table 6, the time to update the status of all ledgers for issueCert is also in the range of 0.91 to 0.98 ms. From the implementation of the algorithm and the experimental results, we confirmed that the proposed infrastructure tends to be able to issue certificates stably.

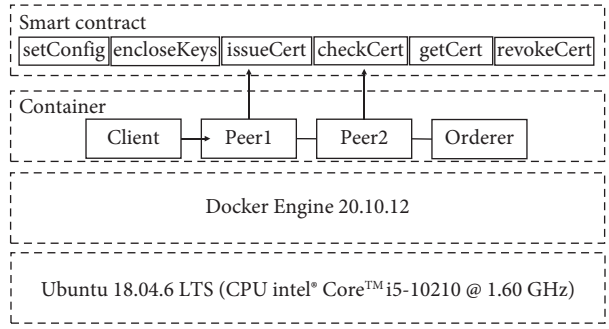


Figure 11: Execution procedure.

Then, we describe the evaluation results of the certificate verification where checkCert is used. From Figure 12, the execution time for checkCert is within a range of approximately 2.9 to 3.4 ms. As shown in Algorithm 4, since checkCert performs a linear search of the CRL to check that the certificate has not been revoked, the execution time is considered to increase linearly. However, our evaluation results showed no clear increase in execution time for as few as 100 CRLs. Once a revoked certificate is listed on CRLs, it is removed from the CRLs when the certificate expires. In other words, the number of CRLs does not continue to increase monotonically, and there is basically an upper limit to the number of CRLs. The upper limit depends on the application to which the consortium CA is applied, but we confirmed that about 100 CRLs have no significant effect on the performance change.

Finally, we describe the evaluation results of the certificate revocation where getCert and revokeCert are used. From Figure 12, the execution time for getCert shows little change, ranging from about 0.5 to 0.6 ms. As shown in Table 2, certificates are stored with its serial numbers as keys, and certificates can be retrieved from the ledger by simply inputting the key as shown in Algorithm 5. As a result, the processing cost of getCert is constant and small. On the other hand, the execution time for revokeCert increased from 0.7 to 2.2 ms. In revokeCert, the inputting certificate is added to the CRLs; hence, it takes the execution time to read and write the CRLs. The number of CRLs increases monotonically in this experiment. Thus, the size of data to be read and written has increased, and the processing time seems to have increased accordingly. However, the execution time is smaller than that of checkCert, which will be executed more frequently, and the number of CRLs is expected to be capped, so performance is not expected to be significantly impacted. From Table 6, in addition, the time to update the status of all ledgers for revokeCert is in the range of 0.67 to 0.75 ms. Compared to the time to update the status of all ledgers for issueCert, the time for revokeCert is smaller. This suggests that the amount of data written to the ledger is related to the time to update the status of all ledgers. Therefore, the time for revokeCert did not show an obvious increase in time, although the amount of data to be written would increase as the number of executions increased.

Table 5: Execution time.

SC	Execution time (ms)					
	1 time	20 times	40 times	60 times	80 times	100 times
setConfig	0.495 ± 0.003	—	—	—	—	—
encloseKeys	4.734 ± 0.017	—	—	—	—	—
issueCert	3.972 ± 0.044	3.876 ± 0.040	4.139 ± 0.491	3.885 ± 0.069	4.206 ± 0.783	3.900 ± 0.057
checkCert	2.946 ± 0.004	2.954 ± 0.014	3.098 ± 0.108	3.353 ± 0.273	3.219 ± 0.246	3.112 ± 0.002
getCert	0.582 ± 0.060	0.566 ± 0.000	0.563 ± 0.000	0.607 ± 0.005	0.641 ± 0.008	0.562 ± 0.000
revokeCert	0.708 ± 0.051	0.774 ± 0.135	1.236 ± 0.011	1.870 ± 0.273	1.967 ± 0.176	2.194 ± 0.025

Table 6: Time to update the status of all ledgers.

SC	Time to update the status of all ledgers (ms)					
	1 time	20 times	40 times	60 times	80 times	100 times
issueCert	0.972 ± 0.005	0.931 ± 0.002	0.953 ± 0.005	0.980 ± 0.016	0.920 ± 0.005	0.914 ± 0.012
revokeCert	0.683 ± 0.004	0.733 ± 0.003	0.702 ± 0.004	0.679 ± 0.002	0.714 ± 0.002	0.759 ± 0.014

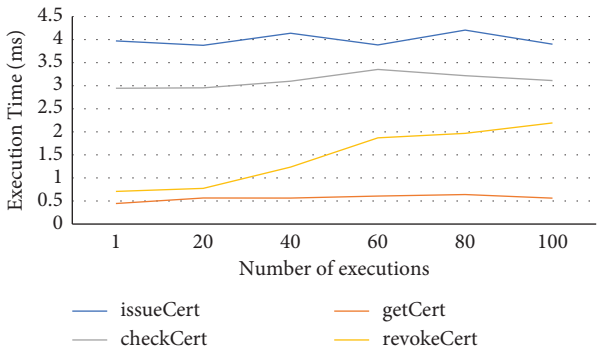


Figure 12: Change in execution time for issueCert, checkCert, getCert, and revokeCert.

8. Related Work

Table 7 shows a comparison of the proposed infrastructure and related studies combining CA-based PKI and blockchain. We reviewed the basic characteristics of them and analyzed them with regard to the fulfillment of the proposed requirements. The basic characteristics of Table 7 are listed below.

(i) Permission type: this indicates whether the study employs permissioned or permissionless blockchain

(ii) Blockchain type: this indicates a type of blockchain systems used in the study

(iii) Private key location: this indicates where the private key used for the certificate is stored

(iv) Certificate generation: this indicates which entity generates the certificate

(v) Certificate verification: this indicates whether the certificate verification process is performed locally or using the SC

(vi) Certificate revocation: this indicates how certificate revocation is performed

(vii) Registration confirmation: this indicates whether a process is used to confirm that the certificate is recorded on the blockchain

Our objective is to propose a consortium CA that can be cooperatively operated by multiple organizations. In contrast, most of the related studies proposing blockchain-based PKI have the objective of resolving the single point of trust of the CA and making the PKI more secure or improving the system for that purpose. Although the differences in the purpose make a complete comparison difficult, we analyzed whether the related studies meet the following four requirements defined in this paper and compared the differences among the related studies.

(i) Req. 1: this indicates whether "Public key certificates are issued without depending on a specific organization." is satisfied

(ii) Req. 2: this indicates whether "All public key certificates issued must be recorded." is satisfied

(iii) Req. 3: this indicates whether "The issuance of a public key certificate cannot be erased." is satisfied

(iv) Req. 4: this indicates whether "Relying parties can confirm that a public key certificate has been issued." is satisfied

The results of the analysis show that the related studies can be divided into five categories from the perspective of the requirements, as shown in Table 7. The first is a category that satisfies none of the requirements. This category includes those that apply blockchain technology for purposes other than certificate issuance (e.g., sharing revocation information or CA policies). Lei et al. [21] propose an efficient certificate revocation scheme in vehicle communication systems (VCS). Ahmed and Aura [22] propose a SC-assisted public key infrastructure (SCP) to manage certificate trust statuses. CAs and domain owners register and publish their certificate policies on the blockchain such that each participant can verify the trust status of certificates. IKP [23] incentivizes CAs to act ethically and report fraud, thereby discouraging abusive behavior. Elloh Adja et al. [24] propose

Table 7: Comparison with related studies of CA-based PKI using blockchain.

	Permission type	Blockchain type	Private key location	Certificate generation	Certificate verification	Certificate revocation	Registration confirmation	Requirements			
								Req. 1	Req. 2	Req. 3	Req. 4
Lei et al. [21]	Permissioned	Custom	Local storage	CA	Local	SC	●	●	●	●	●
Ahmed and Aura [22]	Permissionless	Ethereum	Local storage	CA	Local	SC	●	●	●	●	●
IKP [23]	Permissionless	Ethereum	Local storage	CA	Local	—	●	●	●	●	●
Elloh Adja et al. [24]	Permissionless	Ethereum	Local storage	CA	—	SC	●	●	●	●	●
Certchain [25]	Permissioned	Ethereum	Local storage	CA	Local	SC	○	●	●	○	●
Block CAM [26]	Permissioned	Ethereum	Local storage	CH	Local	SC	○	●	●	○	●
Boyen et al. [27]	Permissionless	Ethereum	Local storage	CA	Local	SC	○	●	●	○	●
CBPKI [28]	Permissionless	Ethereum	Cloud storage	CA	Local	SC	○	●	●	○	○
Wang et al. [29]	Permissionless	Custom	Local storage	CA	Local	SC	○	●	●	○	○
Hwang et al. [30]	Permissionless	Ethereum	Local storage	CA	—	SC	○	●	●	○	○
CertLedger [31]	Permissionless	Ethereum	Local storage	CA	SC	SC	●	●	●	○	○
Yakubov et al. [32]	Permissionless	Ethereum	Local storage	CA	SC	SC	●	●	◐	○	○
Rashid et al. [33]	Permissionless	Ethereum	Local storage	CA	Local	SC	○	●	◐	○	○
Proof chain [34]	Permissionless	Ethereum	Local storage	CA	SC	SC	○	●	○	○	○
Block PKI [35]	Permissionless	Ethereum	Local storage	SC, CA	Local	—	○	●	○	○	○
Li et al. [36]	Permissionless	Ethereum	Local storage	CA	Local	Local	●	●	○	○	○
Proposed scheme	Permissioned	Hyperledger Fabric	Ledger	SC	Local	SC	○	○	○	○	○

○: yes; ◐: partially yes; ●: no.

a new certificate revocation method and status verification scheme. It stores certificate revocation information in a public blockchain and provides a mechanism like a CRL distribution point.

The second is a category that satisfies only Req. 3. The studies in this category tend to utilize the blockchain's tamper-resistance capabilities to ensure the accountability of PKI for specific peers authorized to participate in the blockchain system. Certchain [25] involves an auditing scheme using blockchain for secure SSL communications. It records, publishes, and audits certificate operations such as certificate registrations, renewals, and revocation on the blockchain. BlockCAM [26] proposes a cross-domain authentication model using blockchain. The CA becomes a node on the blockchain, and the CA registers its issued certificates on the blockchain. Boyen et al. [27] propose DPKIT, which eliminates the CA as a single point of failure and ensures transparency in certificate issuance and revocation. Although DPKIT employs permissionless blockchain, auditing requires the cooperation of a dedicated node called DPKIT peer.

The third is a category that satisfies Reqs. 3 and 4. The studies in this category tend to utilize the blockchain technology for ensuring the accountability of PKI for even entities that do not participate in the blockchain system. However, certificates registered in the blockchain are still under the control of a CA, and there is a possibility of omission of certificate registration or registration of fraudulent certificates. CBPKI [28] involves a blockchain-based cloud-based PKI. It aims to leverage the security measures of cloud platforms by offloading certification authority to the cloud. Wang et al. [29] propose a certificate and revocation transparency system to prevent impersonation attacks using fraudulent certificates. Hwang et al. [30] solve the PKI problem using public blockchains that cannot handle numerous certificates using TP-Merkle trees. Kubilay et al. [31] propose a new PKI model with blockchain-based certificate transparency, CertLedger. CertLedger manages the states of all certificates and their revocation status and the set of the trusted CA certificates in blockchain.

The fourth is a category that satisfies Reqs. 3 and 4 and partially satisfies Req. 2. The studies in this category include an additional mechanism to prevent the registration of fraudulent certificates for the third category. Therefore, Req. 2 is partially satisfied. Yakubov et al. [32] propose a blockchain-based PKI management framework for managing certificates. Each CA has a dedicated SC, which allows it to register and revoke certificates. Rashid et al. [33] propose a blockchain-based mechanism for the issuance and management of transparent and secure digital certificates that can prevent CA abuse. The proposed system can solve attacks like Sybil attack, Spoofing attack, and MITM attack.

The fifth is a category that satisfies Reqs. 2, 3, and 4. The studies in this category have an additional mechanism to enforce that all of the certificates are recorded in blockchain for the fourth category. Specifically, the recording process is integrated into the issuing process. Saleem et al. [34] propose a decentralized PKI framework, ProofChain, to improve security. Blockchain miners act as CAs and issue certificates by storing them in the blockchain after each CA signs them. Dykcik et al. propose BlockPKI [35] to reduce the power of individual CAs and to make their actions publicly visible and accountable. A domain owner publishes a request on the blockchain for the issuance of a certificate along with the expected set of CAs. Then, each of the designated CAs performs domain validation and publishes a certificate with multisignature on the blockchain. Li et al. [36] propose a possible solution with new blockchain technology to solve problems like single-point attacks and man-in-the-middle attacks. There is no CA as a third party, and the verifier acts as a CA. When a user registers credential information with the blockchain, the verifier issues the user a certificate for its own server, which is stored in the blockchain. The user can then retrieve the certificate from the blockchain.

In addition to CA-based PKI, a blockchain-based Web of Trust PKI has also been proposed. WoT-based PKI guarantees an identity of a public key without relying on a single point of trust such as a CA. BCTrust [37] proposes a secure communication protocol in wireless sensor networks (WSN). Web of Trust does not use certificates, but authenticates messages by recording them on a blockchain. BlockPGP [38] proposes a blockchain-based framework that manages pretty good privacy (PGP) certificates and key-server infrastructure with high trust. Certificate holders can register and revoke PGP certificates on the blockchain and sign the certificates of others. Blockstack [39] is a blockchain-based naming and storage system . It associates public keys, data, and usernames in a similar manner as PGP using blockchain. DPKI [40] proposes a PKI solution to address attacks coming from a single point of failure in the Industrial Internet of Things (IIoT). DPKI uses a permissioned blockchain, where the participants are all devices in the IIoT network. SCPKI [41] is an alternative PKI system based on a decentralized and transparent design using the Web-of-Trust model and smart contracts on the Ethereum blockchain. Each entity stores its identity in the blockchain, and its authenticity is guaranteed by signatures of other entities. The blockchain stores identities and public keys, and each participant signs these data. Fromknecht et al. propose Certcoin [42], which ensures the association of public keys and identities with a public ledger. Then, Patsonakis et al. [43] improve Certcoin in terms of data size and implement their proposed system in [44]. Qin et al. propose Cecoin [45], which resolves a single point of failure of PKI by recording certificates as currencies to the Bitcoin system. Cecoin has the identity assignment to support delegation of certificate ownership.

The proposed infrastructure is classified into a new category, which satisfies all of the requirements. One major difference from existing research is the capability of storing private keys in a distributed ledger by applying Intel SGX. In most existing studies, a private key of a CA is stored in a CA's local storage. This capability allows the SC to use the private key for generating and confirming certificates.

According to our analysis, none of the existing studies satisfy Req. 1. In the consortium CA, it is important to be able to use a private key cooperatively among CAs participating in the consortium and to prevent fraud by

a specific CA. The proposed infrastructure satisfies Req. 1 by employing blockchain technology and Intel SGX. As a different approach from our proposal, there are some studies that utilize multisignature (e.g., [34, 35]). However, we conclude that they do not satisfy Req. 1 because a CA that creates a multisignature may be able to deny a request based on a CSR.

9. Conclusions

In this paper, we define four requirements of a consortium CA and propose a distributed public key certificate-issuing infrastructure that can be cooperatively operated by multiple organizations. To achieve cooperative operation, the proposed infrastructure encloses CA's private keys in a distributed ledger and enforces the usage of them. We design the proposed infrastructure to meet four requirements and evaluate the fulfillment of those requirements.

In addition, we measured the basic performance of the proposed infrastructure: issuing, verifying, and revoking public key certificates. Through the experimental evaluation, we confirm that the proposed infrastructure can work stably. The proposed infrastructure can issue public key certificates with a processing time of approximately 4 ms and can check that public key certificates are issued by the proposed infrastructure with a processing time of approximately 3 ms.

Data Availability

The data used to support the findings of this study are included within the manuscript.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by JSPS KAKENHI grant no. JP22K17881.

References

[1] CA/Browser Forum, "About the baseline requirements," 2015, https://cabforum.org/about-the-baseline-requirements/.

[2] A. Rech, C. Steger, and M. Pistauer, "A decentralized service-platform towards cross-domain entitlement handling," in *Proceedings of the 2019 IEEE International Conference on Blockchain (Blockchain)*, pp. 455–462, Seoul, South Korea, May, 2019.

[3] K. M. Alam, A. Sopena, and A. El Saddik, "Design and development of a cloud based cyber-physical architecture for the internet-of-things," in *Proceedings of the 2015 IEEE International Symposium on Multimedia (ISM)*, pp. 459–464, Miami, FL, USA, December, 2015.

[4] K. M. Alam and A. El Saddik, "C2PS: a digital twin architecture reference model for the cloud-based cyber-physical systems," *IEEE Access*, vol. 5, pp. 2050–2062, 2017.

[5] R. Roberts and D. Levin, "When certificate transparency is too transparent: analyzing information leakage in HTTPS domain names," in *Proceedings of the 18th ACM Workshop on Privacy in the Electronic Society*, pp. 87–92, London, UK, November, 2019.

[6] Hyperledger-labs, "Hyperledger fabric private chaincode," 2020, https://github.com/hyperledger/fabric-private-chaincode/tree/v1.0-rc1.

[7] B. Laurie, E. Messeri, R. Stradling, E. Messeri, and R. Stradling, "Certificate transparency version 2.0," *RFC 9162*, 2021, https://datatracker.ietf.org/doc/rfc9162/bibtex/.

[8] ITU, "Information technology - open systems interconnection - the directory: public-key and attribute certificate frameworks," 2022, https://www.itu.int/rec/T-REC-X.509/.

[9] A. S. Wazan, R. Laborde, F. Barrère, A. Benzekri, and D. W. Chadwick, "PKI interoperability: still an issue? A solution in the X.509 realm," in *Proceedings of the World Conference on Information Security Education*, Lisbon, Portugal, June, 2009.

[10] Intel, "Intel software guard extensions," 2018, https://software.intel.com/content/www/us/en/develop/topics/software-guard-extensions.html.

[11] V. Costan and S. Devadas, "Intel SGX explained," in *Cryptology ePrint Archive*, 2016.

[12] M. Domb and G. Leshem, "Secured key generation and transmission, using intel-SGX and optical communications," in *Proceedings of the 2019 Third World Conference on Smart Trends in Systems Security and Sustainablity (WorldS4)*, pp. 357–362, London, UK, July, 2019.

[13] Hyperledger, "Hyperledger fabric," 2022, https://www.hyperledger.org/use/fabric.

[14] Hyperledger, "Hyperledger," 2021, https://www.hyperledger.org/.

[15] R. Skuratovskii and A. Kalenyk, "Multisignature with double threshold condition in the blockchain and its application to and strong keys generating," in *Cryptology ePrint Archive*, 2021.

[16] Y. Kaga, "A secure and practical signature scheme for blockchain based on biometrics," *Information Security Practice and Experience*, Springer, Berlin, Germany, pp. 877–891, 2017.

[17] O. Boireau, "Securing the blockchain against hackers," *Network Security*, vol. 2018, pp. 8–11, 2018.

[18] M. Brandenburger, C. Cachin, R. Kapitza, and A. Sorniotti, "Blockchain and trusted computing: problems, pitfalls, and a solution for hyperledger fabric," 2018, https://arxiv.org/abs/1805.08541.

[19] Fpc Team, "FPC without trusted ledger," 2017, https://docs.google.com/document/d/1jbiOY6Eq7OLpM_s3nb-4X4AJXROgfRHOrNLQDLxVnsc/edit#heading=h.sz7cg9d09f71.

[20] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk, "Internet X. Internet X.509 public key infrastructure certificate and certificate revocation list (CRL) profile," *RFC 5280*, 2008, https://datatracker.ietf.org/doc/rfc5280/bibtex/.

[21] A. Lei, Y. Cao, S. Bao et al., "A blockchain based certificate revocation scheme for vehicular communication systems," *Future Generation Computer Systems*, vol. 110, pp. 892–903, 2020.

[22] A. S. Ahmed and T. Aura, "Turning trust around: smart contract-assisted public key infrastructure," in *Proceedings of the 2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE)*, pp. 104–111, New York, NY, USA, August, 2018.

[23] S. Matsumoto and R. M. Reischuk, "IKP: turning a PKI around with decentralized automated incentives," in *Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP)*, pp. 410–426, San Jose, CA, USA, May, 2017.

[24] Y. C. Elloh Adja, B. Hammi, A. Serhrouchni, and S. Zeadally, "A blockchain-based certificate revocation management and status verification system," *Computers and Security*, vol. 104, Article ID 102209, 2021.

[25] J. Chen, S. Yao, Q. Yuan, K. He, S. Ji, and R. Du, "Certchain: public and efficient certificate audit based on blockchain for tls connections," in *Proceedings of the IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pp. 2060–2068, Honolulu, HI, USA, April, 2018.

[26] W. Wang, N. Hu, and X. Liu, "BlockCAM: a blockchain-based cross-domain authentication model," in *Proceedings of the 2018 IEEE Third International Conference on Data Science in Cyberspace (DSC)*, pp. 896–901, Guangdong, China, June, 2018.

[27] X. Boyen, U. Herath, M. McKague, and D. Stebila, "Associative blockchain for decentralized PKI transparency," *Cryptography*, vol. 5, no. 2, p. 14, 2021.

[28] B. Khieu and M. Moh, "CBPKI: cloud blockchain-based public key infrastructure," in *Proceedings of the 2019 ACM Southeast Conference*, pp. 58–63, Kennesaw, GA, USA, April, 2019.

[29] Z. Wang, J. Lin, Q. Cai, Q. Wang, D. Zha, and J. Jing, "Blockchain-based certificate transparency and revocation transparency," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 1, pp. 681–697, 2022.

[30] G.-H. Hwang, T.-K. Chang, and H.-W. Chiang, "A semi-decentralized PKI system based on public blockchains with automatic indemnification mechanism," *Security and Communication Networks*, vol. 2021, Article ID 7400466, 15 pages, 2021.

[31] M. Y. Kubilay, M. S. Kiraz, and H. A. Mantar, "CertLedger: a new PKI model with Certificate Transparency based on blockchain," *Computers and Security*, vol. 85, pp. 333–352, 2019.

[32] A. Yakubov, W. Shbair, A. Wallbom, D. Sanda, and R. State, "A blockchain-based PKI management framework," in *Proceedings of the First IEEE/IFIP International Workshop on Managing and Managed by Blockchain (Man2Block)*, Taipei, Taiwan, April, 2018.

[33] A. Rashid, A. Masood, H. Abbas, and Y. Zhang, "Blockchain-based public key infrastructure: a transparent digital certification mechanism for secure communication," *IEEE Network*, vol. 35, no. 5, pp. 220–225, 2021.

[34] T. Saleem, M. U. Janjua, M. Hassan et al., "ProofChain: an X.509-compatible blockchain-based PKI framework with decentralized trust," *Computer Networks*, vol. 213, Article ID 109069, 2022.

[35] L. Dykcik, L. Chuat, P. Szalachowski, and A. Perrig, "BlockPKI: an automated, resilient, and transparent public-key infrastructure," in *Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 105–114, Singapore, November. 2018.

[36] Y. Li, Y. Yu, C. Lou, N. Guizani, and L. Wang, "Decentralized public key infrastructures atop blockchain," *IEEE Network*, vol. 34, no. 6, pp. 133–139, 2020.

[37] M. T. Hammi, P. Bellot, and A. Serhrouchni, "BCTrust: a decentralized authentication blockchain-based mechanism," in *Proceedings of the 2018 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, Las Vegas, NV, USA, April, 2018.

[38] A. Yakubov, W. Shbair, and R. State, "BlockPGP: a blockchain-based framework for PGP key servers," in *Proceedings of the 2018 Sixth International Symposium on Computing and Networking Workshops (CANDARW)*, pp. 316–322, Takayama, Japan, November, 2018.

[39] M. Ali, J. Nelson, R. Shea, and M. J. Freedman, "Blockstack: a global naming and storage system secured by blockchains," in *Proceedings of the 2016 USENIX Annual Technical Conference (USENIX ATC 16)*, pp. 181–194, Denver, CO, USA, June, 2016.

[40] A. Papageorgiou, A. Mygiakis, K. Loupos, and T. Krousarlis, "DPKI: a blockchain-based decentralized public key infrastructure system," in *Proceedings of the 2020 Global Internet of Things Summit (GIoTS)*, pp. 1–5, Dublin, Ireland, June, 2020.

[41] M. Al-Bassam, "SCPKI: a smart contract-based PKI and identity system," in *Proceedings of the ACM Workshop on Blockchain, Cryptocurrencies and Contracts*, pp. 35–40, Incheon, Republic of Korea, June, 2017.

[42] C. Fromknecht, D. Velicanu, and S. Yakoubov, "Certcoin: a namecoin based decentralized authentication system," vol. 6Technical Report D, pp. 46–56, Massachusetts Inst. Technol, Cambridge, MA, USA, 2014.

[43] C. Patsonakis, K. Samari, M. Roussopoulos, and A. Kiayias, "Towards a smart contract-based, decentralized, public-key infrastructure," in *Cryptology and Network Security*, pp. 299–321, Springer, Berlin, Germany, 2018.

[44] C. Patsonakis, K. Samari, A. Kiayias, and M. Roussopoulos, "Implementing a smart contract PKI," *IEEE Transactions on Engineering Management*, vol. 67, no. 4, pp. 1425–1443, 2020.

[45] B. Qin, J. Huang, Q. Wang, X. Luo, B. Liang, and W. Shi, "Cecoin: a decentralized PKI mitigating MitM attacks," *Future Generation Computer Systems*, vol. 107, pp. 805–815, 2020.

Hindawi
Security and Communication Networks
Volume 2023, Article ID 5614241, 17 pages
https://doi.org/10.1155/2023/5614241

Review Article

Blockchain for Credibility in Educational Development: Key Technology, Application Potential, and Performance Evaluation

Yan Wang, Xin Cong, Lingling Zi, and Qiuyan Xiang

College of Computer and Information Science, Chongqing Normal University, Chongqing 401331, China

Correspondence should be addressed to Xin Cong; chongzi610@163.com

Received 10 November 2022; Revised 21 March 2023; Accepted 3 May 2023; Published 22 May 2023

Academic Editor: Yujue Wang

Blockchain proposes many innovative technologies to establish credible mechanisms in an open environment and therefore, it becomes a promising solution to the problem of credibility in educational development. To better understand the role of the blockchain, we aim to provide an extensive survey focusing on its key technology, application potential, and performance evaluation. First, from the perspective of blockchain characteristics, we summarize its application architecture in educational credibility. Next, we extensively discuss application potential of the blockchain, such as data storage, data sharing, achievement certification, and activity evaluation. Moreover, we investigate the performance evaluation, including basic performance metrics and specialized metrics for credibility. Finally, we analyze the challenges and research trends of blockchain in educational credibility and provide useful insights for future research.

1. Introduction

With ongoing educational reform, many researchers have focused on the issue of trust in the education field. Educational trust is a relationship of affirmative dependent on the educational system arising from the interaction between the trusting willingness of the educational subject and the trustworthy quality of the educational object [1]. Anwar et al. [2] pointed out that in the current educational environment, building trust in education is urgent. It is worth paying attention to the fact that the conventional educational paradigm can hardly adapt to the advancements in science and technology, as reflected in the following aspects. In the past, traditional educational trust relationships were usually based on geography and kinship, with emotional ties as the basic feature, and such relationships were vulnerable to artificial interference, not solid and strong enough, and not scientific enough, which has become a problem for credible educational development. The educational process is implicit and is not conducted under public scrutiny, there can be irregularities, and the results of such education can easily be questioned. In order to address such issues, the establishment of a credible mechanism for education seems extremely necessary. However, in this environment, it is very difficult to establish an open and transparent education credible system without reliable technical support. Considering the previous studies, decentralized technology such as blockchain is introduced to exclude human factors that affect the fairness of the education system and solve the trust crisis in education. Therefore, for educated people, they do not have the ability to assess information on their own and cannot actively choose educational environments and methods that interest them, thus lacking initiative. For teachers, they have no uniform criteria for assessing educated people as a whole, resulting in a reduction. Moreover, educational institutions are not transparent in the process of handling all educational data, and there is no supervisory body, leading to easy leakage of data privacy and reducing data authenticity. Therefore, it is essential to establish an educational credibility mechanism in order to ensure the fairness of the educational process and the effectiveness of the educational results.

Blockchain technology is seen as having great potential for applications in education, assisting in creating a more open and credible education system [3]. It proposes many frontier technologies to create trusted data transaction mechanisms in an open educational environment [4], such as smart contracts [5], asymmetric cryptography algorithms [6], consensus verification [7], and incentive mechanisms [8]. These technologies allow the blockchain to have characteristics such as distributed storage, decentralization, anonymity, and traceability [9, 10]. They break the traditional centralized structure and provide new technical solutions to solve the issue of credibility. However, considering the complexity of the education, the solution to this issue remains very challenging. So, the purpose of this paper is to explore how the blockchain can build a credible mechanism in an open educational environment.

The main contributions of this paper can be summarized as follows. (1) We summarize the application architecture of blockchain in educational credibility, including core technology and attributes. We highlight the core technologies, such as digital signature, consensus mechanism, encryption algorithms, and smart contracts. (2) On this application architecture, we demonstrate the application potential of blockchain in four aspects and for each aspect, we analyze current credibility issues in education and how blockchain can help address them. (3) To evaluate the performance of the blockchain-based systems, we provide basic performance metrics and specialized metrics. The former evaluates the important performance of the blockchain system itself, while the latter gives the unique evaluation method for assessing credibility.

The rest of the paper is structured as follows. Section 2 provides the architecture of the blockchain, Section 3 demonstrates educational application of the blockchain, Section 4 presents performance evaluation, and Section 5 is conclusions.

2. The Application Architecture of the Blockchain

Blockchain can be described as an immutable ledger that records data in a decentralized manner, which enables entities to interact without the presence of a centrally trusted third party [11], exploring the blockchain application architecture from a typical blockchain application in an educational environment. MOOCsChain [12] is the blockchain application on the MOOC platform, which consists of five main parts, registration authority (RA), MOOCs providers (MPs), end-users (EUs), blockchain (BC), and data storage servers (DSs). RA is mainly responsible for handling all platform registration requests and providing public and private keys for authorized users. EU is a port for users to use the platform and participate in the course. MP is a course content provider, and each MP is an independent entity that can communicate with the storage server. BC records key materials of learners using smart contracts and provides a decentralized storage environment. DS stores data through a distributed storage system to protect the limited storage capacity of BC. Publication Chain (PubChain) [13] mainly relies on the blockchain system and IPFS system. The blockchain runs a distributed consensus protocol to maintain the data on the chain, and participants interact with the blockchain when running activities on the PubChain.

Considering the particularity and complexity of application scenarios in education, the blockchain technology architecture can be divided into three parts, as shown in Figure 1. In the first circle, the disordered education data are added to the chain structure and stored according to the structure of Merkle tree; these nodes made up the bottom layer of the blockchain structure P2P network [14]. In the second circle, the core technology of the main applications of blockchain contains digital signature, consensus protocol, smart contract, and asymmetric encryption algorithm, which improves the legitimacy of the educational material [15]. In the third circle, benefiting from the core technology of the second circle, blockchain will have some attributes such as traceability [16], authenticity, anonymity, and security, which can be useful in educational scenarios such as certificate verification, online learning platforms, and lifelong learning records.

2.1. The Core Technology. The education filed mainly concentrates on the application of the core technology in blockchain, such as consensus verification, asymmetric cryptography algorithms [17], digital signature and smart contracts, which have their unique properties and complement each other to cooperate in educational scenarios. While ensuring the authenticity of the educational data, they help promote the construction of a credible system for education.

2.1.1. Smart Contract. A smart contract is a computer protocol designed to disseminate, validate, and enforce a contract in an informative manner, a piece of computer code that constitutes a program. It plays an important role in educational applications. In the first step, two or more users involved in educational activities agree to formulate their common opinion into a smart contract; in the second step, this smart contract is broadcast and stores to the block nodes in the framework through the blockchain network; in the third step, the successfully constructed smart contract waits for the conditions to be met and then automatically executes the contents of the contract. It is worth noting that not all blockchains have smart contracts, such as beacon chains. Blockchains that do not have smart contracts differ in the way they solve problems. Smart contracts are the unique existence of the blockchain technology that can convert the coding of data interactions into contracts and related documents in the traditional sense [18]. Smart contract provides a more fair and equitable method of transaction with transparent data, while minimizing interaction of parties in a decentralized manner [18]. These transaction data are traceable and irreversible and can be automatically executed according to the provided terms without the involvement of any third-party [19], enabling the sharing of data.

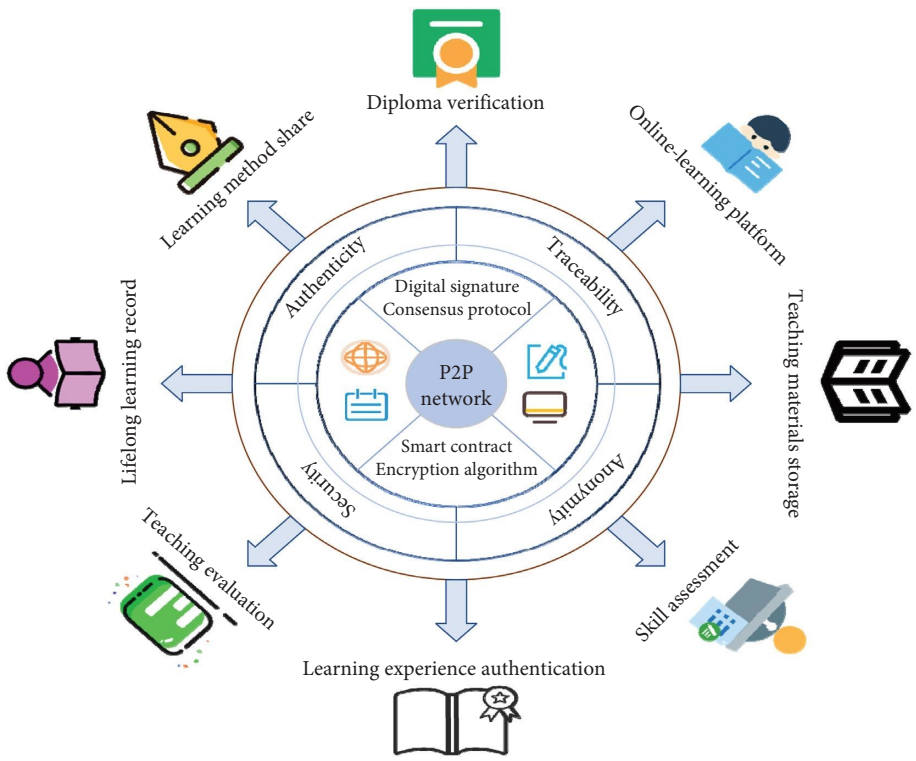


Figure 1: The application architecture of the blockchain in the educational credibility.

As can be seen from Figure 2, the education role inputs education data to the smart contract, which has a contract state, contract value, and contract code, and the contract is executed automatically after getting the input data. The corresponding output is obtained according to the contract content. There is no third-party involvement in the process, and the contract content will not be changed in the middle of the process to ensure the consistency of the result data. The execution of smart contracts has a huge impact on the blockchain technology [20], and all participants carry out contracts in accordance with the same standard to achieve maximum fairness and credibility.

2.1.2. Consensus Protocol. The main purpose of the consensus protocol is to enable decentralized network nodes to reach an agreement and complete the consensus verification. In a central and organizational body, all decisions are judged by selecting a highest-priority role [21], which is highly subjective and is unfair and lacking in credibility. However, the outcome is determined by all participating node and are subject to their interests in a distributed network. This process is known as consensus [22]. The specific consensus verification process is illustrated in Figure 3.

As can be seen from Figure 3, the main players in educational activities contain students, teachers, schools, and institutions who join together in a smart contract to choose the appropriate consensus protocol for their desired educational activities. For example, PubChain uses the PoA consensus protocol in federated chains and the PoW consensus protocol in public blockchains [23]. The consensus mechanism is that all nodes on the blockchain communicate consistently. When the educational data on a block change, all users will receive a notification and update their data status in time, solving the problem of synchronizing educational information data in an educational environment, and all its behaviors will be supervised [24]. The common consensus protocol is listed in Table 1.

2.1.3. Asymmetric Encryption Algorithms. Asymmetric encryption algorithms use key pairs, public and private keys to protect the information of users in the blockchain network [40]. The public key and the private key are generated simultaneously and play a decisive role in the subsequent creation, change, or view of the information in the block [41]. The user uses the public key to encrypt the data information, determining the authenticity of the information. Then, the only authorized user can use the private key to decrypt and access to obtain data. The execution flow of the asymmetric encryption algorithm is depicted in Figure 4.

As can be seen from Figure 4, students can encrypt their educational data and personal data using asymmetric encryption algorithms. If a teacher, school, or employer needs to access the student's data, it needs to be authenticated by the public key given by the student, and after the authentication, a series of educational activities can be carried out. Cryptography is one of the primary tools for ensuring data security [42], the most widely used asymmetric encryption algorithms, such as the RSA algorithms [43, 44], run slowly, have open methods, and encrypt data quickly, but the management of private keys is not secure enough. The DSA algorithm [45] has slow running speed and faster performance compared to RSA [46] algorithm, which is only

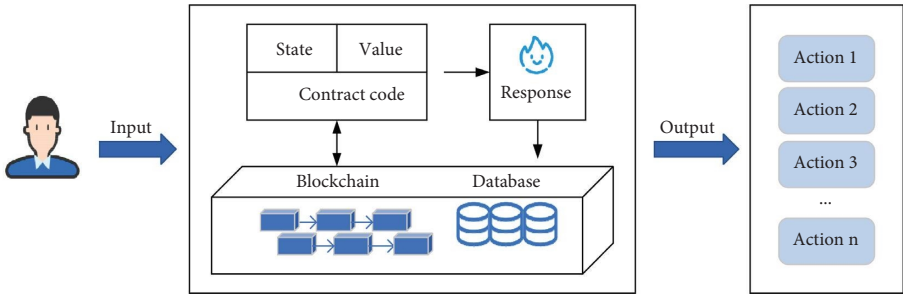


Figure 2: The process of executing contract in educational application.

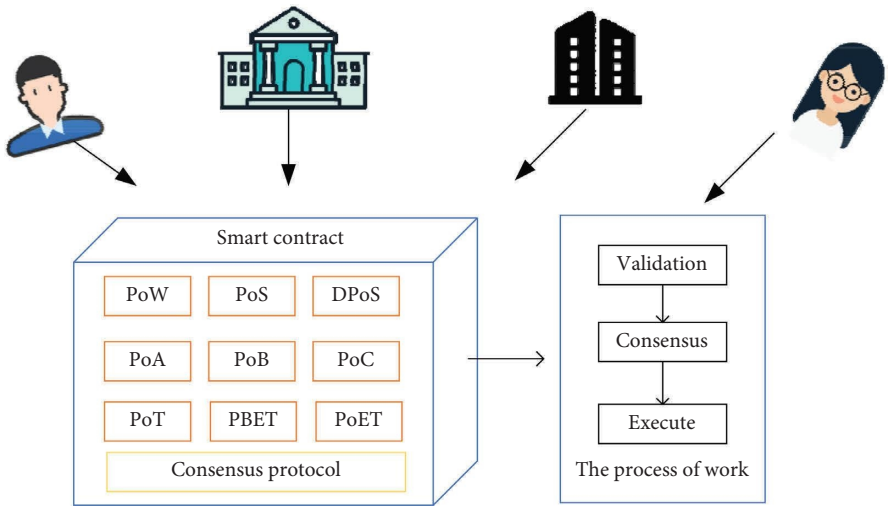


Figure 3: The process of consensus protocol in educational application.

capable of digital signatures, not for data encryption or decryption. Elliptic curve digital signature algorithm (ECC) [47, 48], runs fast, can use smaller keys, more efficient, but long operating times for encryption and decryption. The PTFT algorithm [49], fast running speed, high security, difficult to attack, but one-way strategy, and complex decryption process.

2.1.4. Digital Signature. A digital signature (also known as a public key digital signature) is a string of numbers that can only be generated by the sender of a message and cannot be forged by anyone else, and it is a valid proof of the authenticity of the message sent by the sender. It is an ordinary physical signature, similar to the one written on paper, but implemented using techniques in the field of public key cryptography, used to authenticate digital messages. A set of digital signatures usually defines two complementary operations, one for signing and the other for verification. Digital signatures are applications of asymmetric key cryptography. When educational data are stored, it is encrypted using asymmetric encryption algorithms, at which points a digital signature is used in an act similar to "stamping." The application of digital signatures mainly adds a layer of protection locks to educational data, verifies the user's identity information, traces the authenticity of the information source, prevents data from being tampered with and forged, increases the credibility of the information, and creates a more transparent and secure educational system.

2.2. The Attributes of Blockchain. The key attributes of blockchain applications in education filed are traceability, authenticity, anonymity, and security, which merge and complement each other and work together in establishing credible mechanisms.

2.2.1. Traceability. Traceability is due to the fact that all transactions on the block are sorted chronologically, and the previous block and the next block connected to itself can be found between blocks by index values. The index value on the block uses a one-way hash function, and there is no direct necessary connection between input and output. In other words, the input cannot be determined by just giving the output, so that the origin of the transaction data recorded in the block and the source of the data can be traced.

2.2.2. Authenticity. Blockchain is decentralized networks without the control of a central authority, and block nodes supervise each other to strictly prevent tampering attacks by malicious nodes. When a new node record is created, it is

Table 1: Consensus protocol.

Protocol name	Description	Advantages	Disadvantages
PoW [25–27]	Proof of work protocol, solving problems through miner mining [28]	Good performance against malicious node attacks	Takes a certain amount of energy [29]
PoS [30]	Proof of stake protocol, nodes with access to benefits to solve the problem [31]	Reaching consensus takes a short time	Hard forks are prone to occur
DPoS [32]	A special case of PoS, forming a consensus group to solve problems through the public interest	Propagation speed is fast Higher throughput	Elections are needed to determine the consensus group, and only a small number of nodes are elected
PoA [33]	Proof of authority protocol to publicly certify all document processes with trusted nodes	Small scale High credibility	Preventing node collusion requires user supervision
PoB [34]	Burn the proof protocol and select the final result through an algorithm	Fewer validators More efficient	Requires a lot of resources to test
PoC [35]	Capacity proof protocol with the hard disk as the consensus participant	Low cost addresses global trust and security	High energy consumption requires sacrificing node performance
PoT [36]	Proof of trust protocol, through the incentive mechanism, the node gives honest verification results	High throughput, low energy consumption	Nodes are likely to commit malicious behavior
PBFT [37]	Practical Byzantine fault tolerance algorithm, malicious nodes are not higher than 1/3 of the total	At the same time guarantee safety and activity	The node must be deterministic and must start execution from the same state
PoET [38]	The time it takes to prove consensus algorithms, typically used in permissioned blockchain networks, to determine mining rights on the network	The cost of participation is low, and more nodes can be easily joined	Requires specific hardware and is not suitable for large-scale applications
RAFT [39]	An algorithm that implements distributed consensus and is mainly used to manage the consistency of log replication	Easier to understand and apply to real systems	Only faulty nodes can be accommodated, not evil nodes

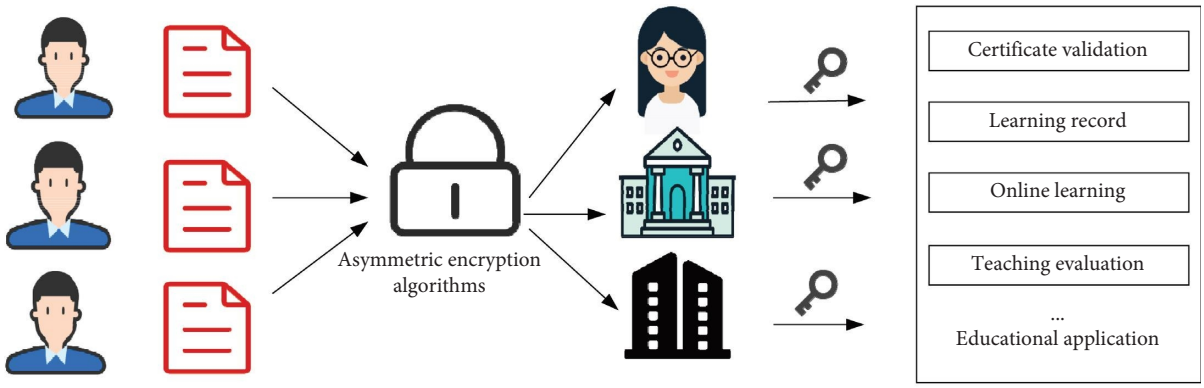


Figure 4: The process of encryption in educational application.

first verified by network nodes and then added to that chain. The data verified by the nodes will not be modified again and remain in its original data state.

2.2.3. Anonymity. The block nodes are all peer nodes with the same priority and structure in the network. On a technical level alone, the identity information of each block node does not need to be disclosed or verified, and information transfer can be done anonymously. The user access to the data is hidden and the information sharing process is also delivered anonymously and encrypted.

2.2.4. Security. The modification network cannot be controlled unless you have control over 51% of all data nodes, which makes the blockchain itself relatively secure from human subjective data changes. Only users with public keys can access and read the data because it is encrypted and stored using a highly secure asymmetric encryption process.

The previous attributes are the ones that education data will have when education applications are combined with the blockchain technology. For example, in the higher education certificate authentication system, the certificate data can be guaranteed to be real and safe after being processed by the blockchain technology and the source of that certificate data can be traced. The credibility of the certificate obtained through such an authentication process is greatly improved.

2.3. The Types of Blockchain. There are three types of blockchains, which are as follows:

(i) Public blockchain: each node on the public chain can freely join and exit the network and participate in the reading and writing of data on the chain, interconnecting with a flat topology when reading and writing, and there is no centralized server node in the network.

(ii) Private blockchain: the right access of each node in the private chain is controlled internally, while the read access can be opened to the public selectively on demand.

(iii) Consortium blockchain: each node of a federated chain usually has a corresponding physical institutional organization that is authorized to join and exit the network. Each institutional organization forms a stakeholder alliance to maintain the healthy operation of the blockchain.

The core difference between these three types is the degree of openness of access or decentralization. In general, the higher the decentralization is, the higher the trust and security and the lower the transaction efficiency. Usually, depending on the characteristics of the educational application itself, a type with a higher degree of adaptability is chosen based on the actual situation. In educational storage applications, the more decentralized type will be preferred, while educational assessment, authentication, and sharing applications will use the more efficient type.

3. Educational Application of the Blockchain

Blockchain technology has infinite possibilities for a wide range of application in the field of education [50]. Through a review of published papers, the application in educational credibility can be divided into four areas, including educational data storage, educational data sharing, educational achievement certification, and educational activity evaluation, as shown in Figure 5.

3.1. Educational Data Storage. In the field of education, various educational activities are emerging and more data are generated in the process of the activities. When traditional methods are used to handle data and manage process, there are problems in terms of efficiency and security of data storage. In terms of efficiency, since education is still largely controlled by institutions that provides quality, credibility, governance, and administrative functions [51]. However, many educated people have learning data at different stages of the educational process, thus these data are stored independently in different institutions, so this can affect the efficiency of querying the data. In terms of security, institutions generally store educated peoples' learning data in the form of a central database for unified management, which is singularly uncontrollable. Once there is a problem with the database, there is a great risk that the stored data will be tampered with or even lost. In addition, the lack of

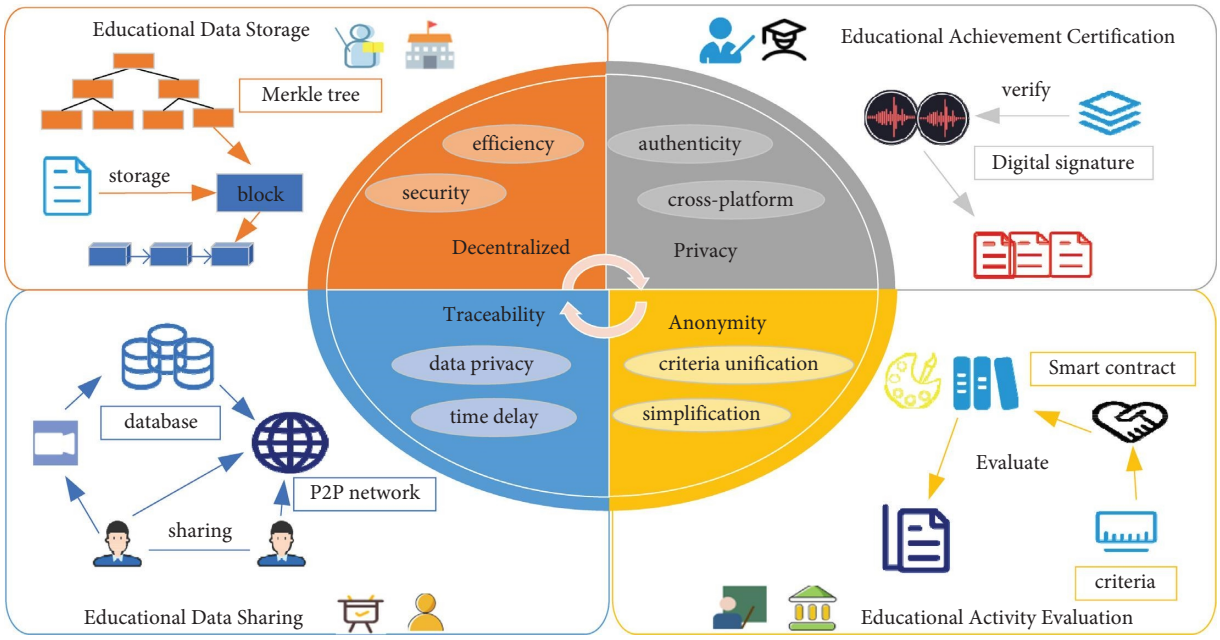


Figure 5: The application of the blockchain in educational credibility.

a supervisory and management body during the data storage process, and thus, the inability to guarantee data constancy, indicated that the privacy and security issues are of great concern. Therefore, how to store data in an efficient and secure way is a question we need to consider.

Blockchain offers a possible technology to solve the previous problems. For data storage efficiency, due to the limited block space, educational data updated by students are added to the blockchain in the form of blocks. The blocks store extremely important and effective educational information, such as students' basic personal information and educational data. Subsequent transactions can be made directly on the blockchain when educational activities are carried out, and the entire transaction process is guaranteed to be trusted. At this time, a hash index value is generated on the block, which can quickly locate where the block is located on the chain, and then efficiently query the data carried by the block, reducing the time cost of data query and analysis. For data security, educational data are stored using asymmetric encryption methods with digital signatures, and the generated key pairs can be accessed by authorized users who have public keys, thus avoiding abusive tampering of data by malicious nodes [52]. The decentralized nature of the blockchain technology allows the stored data to be free from the control of a central database, that is, distributed storage gives the students themselves full control over the management of the data. Since blocks are equivalent to peer nodes in the blockchain network, data interactions on blocks are always under the common and strict supervision of other blocks on the chain, ensuring the transparency of stored transactions and thus reducing the possibility of data privacy leakage.

Currently, researchers have applied the blockchain technology in data storage, such as student credit management [53], achievement management [54], and career data management in nonformal education [55]. Liang et al. implemented PDPChain [56] for secure storage of personal education data, and the blockchain network in the framework guaranteed the trusted storage of the private data by using a consortium chain. The encrypted ciphertext hashes are stored in a smart contract in the consortium blockchain, and transactions with consistent communication are sent to the network using the RAFT consensus mechanism. After the cryptographer verified the digital signature, the transactions are packaged, blocks are generated for sorting, and finally, the blocks are stored in the blockchain network intact. After this process, the personal education data stored in the blockchain network is safe and secure, and data transparency is truly achieved. Many scholars view the blockchain as the underlying architecture that stores all data transaction records in a ledger. Rooksby and Dimitrov [57] used the blockchain to register to determine ownership of intellectual property, preserve academic transcripts, and establish a more scientific storage model. Kosasi et al. [58] see blockchain as a digital system that offers tremendous potential for the storage of student educational records in the use of the higher education. Data privacy and security are ensured through unique asymmetric cryptography algorithms that ensure the storage of student records and credentials [59, 60]. The blockchain technology, with its unique advantage of data immutability, stores students' certificates of achievement that can accurately predict the future based on experience and helps students develop personal plans with the help of various algorithms [61]. Turkanovic et al. [62] proposed an ecosystem for managing digital micro-credentials (EduCTX), a global credit platform for higher education based on the blockchain technology. The main function of this platform is secure transfer and accumulation of credits. Students can store the credits they have earned during their studies in the system and when changing

institution they do not have to worry about losing or tampering with their data. Ocheja et al. [63] presented a method to save learning records, and the scheme's is to securely store students' learning data. When learners switch to a new learning environment, they can take all of their learning records with them, ensuring the immutability and security of the educational data. Awaji et al. [64] proposed a secure system for achievement records, which attempts to store students' achievement records efficiently, encrypted in the form of blocks that can be easily located for queries and improved the efficiency of students' searches. Li and Han [65] developed the storage platform, a blockchain-based storage, and sharing scheme for educational records (EduRSS) to accomplish security and privacy protection of educational record storage.

Obviously, the blockchain technology can improve the efficiency of education data storage and create a more secure data storage environment, thus strengthening the trustworthiness of education data storage and ensuring the consistency and consistency of data in the storage process. Nevertheless, data overload is an issue we need to further study in an environment where data are highly trusted to grow rapidly.

3.2. Educational Data Sharing. In the environment of the Internet era, data sharing has become a major trend that can benefit multiple participants. In the field of education, we divide data sharing into two aspects: educational resource sharing and educational data sharing. Educational resources include various forms of resources, such as teaching software, teaching videos, and teaching environments. Educational data mainly refers to all data generated by educated people throughout their educational activities. Both are core components of educational activities. Traditional forms of data sharing are point-to-point transfers by data producers, which have high time delays in transmission and do not guarantee data privacy. Considering the time delay, in the context of modernization of education, information about educational resources and educated people is commonly shared among multiple parties in educational activities. Data producers provide the prepared data to the shared recipients, and the delivery process requires significant time costs. Moreover, in the case of sharing core educational data, the sharing process takes too long and is prone to security problems of data loss. In terms of data privacy, due to the wide application of artificial intelligence and big data technologies, data sharing transactions in education are becoming more and more frequent, and the issue of data privacy leakage is becoming more and more prominent. During the sharing process, the privacy and security of data can be damaged by a large number of users, and the availability of data can be greatly reduced. At present, vigorously promoting education data sharing has been a national strategy to promote the development of education information, and we need to reduce the sharing delay and improve data availability.

Blockchain network topology and anonymity protection technology can be used to solve the aforementioned issues. For time delay, the P2P network [66] topology contains a distributed structured topology (DHT) [67], which is a massive hash table maintained collectively by all nodes. This effectively reduces data latency. When users need to access data, they can directly query the hash index value for access, avoiding the intermediate transmission link, and the decentralized nature of the blockchain technology reduces the response time of access. For data privacy, data are encrypted and packaged in blocks, which is then connected in a chain to form a distributed ledger system. Also, anonymous technology can help to safeguard privacy [68]. Only authorized users have access to the private key for decryption, while educational data are keeping encrypted with public keys using asymmetric encryption methods. The original educational data are not shared directly on the blockchain, thus preventing the privacy of core data from being compromised and improving the security of sharing.

Currently, many scholars have developed a number of open-source platforms for data sharing. Gao [69] developed a platform of top-notch educational materials for universities and institutions. It compiles large number of educational resources that can be quickly accessed by the educated and used for self-study. PubChain, a decentralized distributed open access publishing platform based on blockchain and IPFS peer-to-peer file sharing system, was designed and implemented by Wang et al. [13] PubChain used the blockchain technology to confirm the registration of ownership of papers, track indexing, and be cited. When an author uploads his or her paper to PubChain, the paper was timestamped and registered as a permanent record. Compared to existing centralized publishing platforms, PubChain made papers freely available to everyone, eliminates the undesirable effects of information silos, and has the potential to become a unified database for sharing and recording papers globally. The sharing of smart education courses in institutions is an important way to improve the quality of individual students [70], and implements such an architecture for wireless communication requires prioritizing the blockchain technology that provides security and data transparency [71]. Using the blockchain technology to visualize student data to display learning outcomes addresses data transparency in the sharing of outcomes under the influence of teaching or administrative processes [72]. Various online education platforms provide a broad Internet environment for sharing multimedia learning resources, and the blockchain technology needs to be used to address the risk of decreasing trust in the process of resource sharing [73]. Gilda and Mehrotra [74] broke the conventional practice of sharing student data in paper form by using the blockchain technology to build a framework of trust and authorization to complete the overall assessment of students using the data obtained. Zhao et al. [75] proposed a sharing system for digital education resources, allowing educators and educational institutions to share teaching videos. Sharing records are not seen by others, and once a video is

published in the system, it cannot be changed again, ensuring the accessibility and authenticity of digital resources. Han et al. [76] created a storage authentication structure that allows users to confirm the accuracy and integrity of outcomes from shared data using the blockchain technology. In addition, there are research results sharing platforms [77], skills sharing platforms [78], and school sharing online education platforms [79].

In summary, the sharing mechanism constructed by using the blockchain technology effectively reduces the time delay of data sharing, enhances the privacy and availability of shared data, and facilitates the establishment of a credible educational data sharing mechanism. However, in the process of data sharing, the confirmation of data resource ownership is the next step we should consider, which requires the support of multiple parties such as laws, policies, and standards.

3.3. Educational Achievement Certification. Certificates are the most reliable foundation for confirming students' own valuable capabilities in the wave of global education. It is a concise and direct reflection of the student's personal abilities through degree certification, learning record certification, skills certification, and results certification, and it is also a reference for corporate background checks during the job search process. Thus, verifying the authenticity of certificates such as diplomas or achievements can be a long and expensive process, and there is a risk of certificate forgery. Currently, students' learning experiences on different electronic platforms cannot yet be integrated or recognized, reducing students' motivation to learn online. In terms of authenticity, since certificates are usually paper-based, the data can be easily modified during storage and their privacy cannot be guaranteed. In addition, if the issue agency ceases to exist, the authenticity of the certificate will not be verified, which will lead to problems of diploma fraud and forge certificates. Therefore, educational certification is a matter of everyone's rights, and it is urgent to solve the related issues.

Thanks to the development of smart contracts and consensus protocol, tamper-proof and traceability features are considered as the best solution to the previous mentioning issues [80]. For cross-platform verification, learning platforms built with the blockchain technology can automatically issue digital certificates according to learning outcomes, and digital certificates are increasingly popular with the public because of their small size and easy preservation. Digital certificates integrate users' all cross-platform learning experiences and store them using the blockchain technology. Blockchain store user data in a distributed manner with decentralized features, uses smart contracts to incorporate users, enterprises, schools, and other roles into the platform, and adopts consensus protocols to form a multirole and certification platform. For data authenticity, educational data are packaged in blocks with encryption algorithms that will not be modified or deleted, and then, the blocks are added to the chain in chronological order to facilitate subsequent verification of the data by the employers or other departments to trace the authenticity of the data source.

There are some educational certification platforms based on the blockchain technology that provide users with one-stop certification solutions. Tian et al. [81] redesigned the expandable framework for validating the integrity and validity of educational digital evidence. The main execution process of the framework is that content providers submit educational digital evidence in their possession, and the framework records the documents in transactions, with multiple transactions stored on a block that contains the public key for preventing tampering with the evidence and tracking the corresponding documents. It uses the PBFT consensus mechanism to effectively guard against malicious and faulty nodes. When a new block is generated, the block is broadcast to other nodes. When each node receives the block, it verifies all transactions in the block by comparing the Merkle root in the block with the Merkle root in the node. It also verifies the authenticity and validity of the digital evidence of education and creates a fair and credible educational environment. Multimedia learning resources are becoming more and more abundant, and students are earning more and more certificates for studying on various online platforms [73]. The insecurity of digital education certificates makes students' abilities not well proven. By introducing blockchain, a technology that combines public and private chains using specific smart contracts, the verification of certificates on various platforms is realized [59]. Using certificate authorization services and transactions in the Hyper Ledger framework to achieve transparent information sharing between universities and enterprises, the information symmetry between students' skill achievement information and enterprises' recruitment demand is realized [82]. Sanni and Apriliasari [83] proposed a blockchain technology authentication system that can protect data rights from interference, and all data stored in the education system is secured. Due to the decentralized nature of the blockchain, the trust of parents, teachers, and other parties in the education system will be increased. Han et al. [84] suggested combining the blockchain technology and certification methods, using smart contract to integrate multiple players such as departments, colleges, universities, government agencies, and businesses to certify the formal educational achievements of educated people. Arenas and Fernandez [85] proposed Credence Ledger, a blockchain solution that decentralizes the authentication of academic credentials so that employers can quickly confirm the authenticity of this information. To create a student central approach to achievement certification in an open learning environment, Awaji and Ellis [86] developed a blockchain technology-based achievement certification system using the PoW consensus protocol for certification, which certifies grades that can be recognized by third-party entities. Bandara et al. [87] used a high trust level of the blockchain technology to build a distributed and secure collaborative certification database to verify the results of informal education in parallel with the university, partner institutions, and regulators. Alshahrani et al. [88] proposed a framework for higher education certificate certification based on the

blockchain technology, in which education certificates are stored and the authenticity of higher certificates is verified using DPoS consensus protocol, and the certification process will become more convenient and credible. Similar educational certification precautions include credit certification [89], degree information certification [90, 91], and diploma certification [92].

In generally, the blockchain technology can rely on its own decentralized distributed storage management to achieve the authentication of cross-platform educational data. It can also guarantee the authenticity and validity of the educational data through immutability and data traceability. Meanwhile, it also can be used to create credible education achievement verification systems. The next issue we must address is how to construct a greater security consensus smart contract to enhance credibility because the authentication process requires the inclusion of multiple players such as schools, employers, and regulators.

3.4. Educational Activity Evaluation. In order to better improve the quality of education, when the educational activities are completed, they are evaluated accordingly. Students can evaluate the teacher's teaching, and teachers can also evaluate the students' abilities. On the one hand, the assessment behavior requires obtaining data from multiple parties and then synthesizing and processing the data. In general, this process is opaque and costly in terms of human and material resources. On the other hand, assessment criteria are usually defined by the high-priority roles involved in the assessment activities. However, the evaluated roles are largely passive [93]. In addition, the evaluation criteria vary from platform to platform, so the evaluation results are difficult to be recognized. In this case, it is very difficult to make an objective and comprehensive analysis for the educated people, therefore, a unified evaluation standard is needed due to the complexity of the evaluation process.

The blockchain technology includes consensus protocol and digital signature, which can provide new ideas and technical support for the problems in the educational activity evaluation process. In terms of process simplification, the roles involved in the evaluation of educational activities can clear know the educational data of all the evaluated objects on the chain, and can give the evaluation results directly through the educational data in a fair and open manner. During the evaluation process, the consensus protocol is equivalent to a broadcasting role. When the evaluation object gives the final result of the evaluated object, the blocks on the chain and all participants of this educational activity will be notified through the consensus protocol to ensure consistent communication information of all blocks. The entire process of evaluating educational activities is transparent, and the behavior of evaluation participants is monitored throughout. This simplifies the complex educational activity evaluation process, which originally requires multiple participants, upper-level discussions, and collective evaluation opinions, to the evaluation results given by the roles involved in the evaluation activity through a consensus protocol. In terms of standard unification, educated people can have different learning activities in multiple platforms, and different platforms have different criteria, and there is a lack of unified evaluation recognition criteria for the complete educational activities of the educated people. Evaluation can only be done by integrating the results of educational activities from different platforms, but such an evaluation behavior lacks real and completed data basis, and the obtained evaluation results do not have strict credibility. The consensus mechanism provides a powerful tool for the unification of evaluation criteria, which can objectively evaluate the results obtained by users after cross-platform educational activities. The data tracing function of the blockchain can record users' activity behaviors and enrich the details of the evaluation process, thus improving the reliability of assessment results.

There are several examples of the blockchain-based evaluation systems, such as Li et al. [94] developed a skills assessment system that enables teachers to assess student skills and teaching effectiveness based on the learning data in the system to create a fairer, healthier, and more open e-learning and online education environment. For any educational institution on the system can create and deploy course credit generation contracts on the blockchain, which contain information such as test scores, learning hours, and commenting behavior as a basis for automatically assessing users' specific course credits, simplifying the process and preventing data from being undisclosed and opaque. Lizcano et al. [95] see blockchain as the technology used to manage teaching content and student competencies by consensus among students, teachers, and employers, bridging the divide between academia and the world of work once and for all. The assessment of student learning and professional competencies [96] are performed automatically with the same criteria set by all parties involved in the assessment activity [97]. Widayanti et al. [98] showed that the assessment process using the blockchain technology as the underlying structure disrupts the traditional educational model and the results of automated assessment are more convincing. Zheng [99] created a learning assessment system that evaluates students in an anonymous way and the system is able to obtain the results quickly, ensuring that the results are objective and fair. Zhao et al. [100] proposed a blockchain technology-based student competency evaluation system that focuses on monitoring students' educational activities, analyzing learning data, and developing unified evaluation criteria through a PoA consensus protocol to objectively and comprehensively evaluate students' personal skills demonstrated in educational activities and give reference to students' future job search directions. Stepanova and Erins [101] proposed a career growth data evaluation model, which records the learning activity experience of an educated person in nonformal education and sets common criteria to assess the occupational competence of that user through a consensus protocol. Wu and Li [78] upgraded the personal skills competition model using the blockchain technology to analyze and unify the existing evaluation criteria and simplified the evaluation process. The assessment results were given directly through the students' skill operations on the operating system of digital education,

which greatly improved the efficiency of activity evaluation and the accuracy of the results. Jirgensons and Kapenieks [102] discussed digital certificates and how the data traceability and decentralization of the blockchain technology can be used to develop unified recognition criteria to improve the credibility of the evaluation of educational activity certificates.

Generally speaking, the blockchain technology provides a solution to the current evaluation model with complex processes and lack of uniform criteria in educational activities. In addition, the traceability and authenticity of data enhances the credibility of evaluation results. Due to the complexity of the education field itself, most of the evaluation activities still require human intervention. How to use blockchain combined with artificial intelligence technology to train evaluation models and continuously optimize them is the next direction we need to study.

4. Performance Evaluation

Evaluation metrics are mainly used to measure the overall performance of the system application. Inspired by the software quality metrics, the evaluation metrics for studying and analyzing the blockchain technology in building a trustworthy mechanism in the education field from the perspective of expected target results should consider both the performance situation of the blockchain technology itself, the basic performance metrics; and the performance situation after application in the education field, the characteristic metrics. The basic performance metrics are response time, cost, throughput, efficiency, and reliability; the characteristic metrics are scalability, consistency, maintainability, real-time, and processability. We give the corresponding calculation formulas on different evaluation indexes, which are mainly derived from the common system performance evaluation criteria calculation rules, obtained after many practical studies. The performance of the system is accurately measured by mathematical methods. As the basis of [103], formula (1) similar to calculate of the latency time. According to the mathematical formulas, due to the same of underlying calculation logic, uses the cost required for a single block product the total number of blocks, getting formulas (2)–(4). Similarly, based on the [104], the calculation of efficiency, formulas (5)–(7) mainly uses response time division to the cost. The cost includes CPU, memory resource, and time.

4.1. Response Time. It refers to the time required from the start of a block transaction until the result is recorded on the blockchain after the transaction is closed. Define at the time of t_s transaction starts, at the t_e moment the transaction is recorded on the blockchain, in a period of time T, the total number of transactions is T_s, written as follows:

$$Ts = \sum_{te \in (0,T) \& (te-ts) \leq T} ts, \tag{1}$$

where t_s is the count value, and when the start time and end time meet the conditions, t_s is 1, otherwise 0. When education data are added to the chain in the form of block storage, the shorter the response time, the more efficient the storage is demonstrated. In data sharing, the shorter the response time, indicating that the user accesses the block data quickly, and the data sharing transaction is convenient and rapid.

4.2. Cost. It refers to the consumption of the blockchain in the process of executing the application. Public chains generally encourage nodes to synchronize information and ensure data security through a token mechanism. This way of storing data will make the cost of educational applications using public chains increase, but this increase is acceptable relative to the benefits it brings. Other types of resource cost consumption are similar for the three types of blockchains. There is a certain amount of loss in the process of generating a block's transaction, and different cost types consume in different ways. When the consumption is a resource, assuming that the energy cost per unit consumed is E_s, the time consumed by a single transaction is T, and a single transaction refers to the time that the block lasts from generation to being added to the blockchain. Thus, the cost E calculation can be focused on the use of the central processing unit (CPU), written as follows:

$$E = Nn \cdot Es \cdot \int_0^T CPU(t)\mathrm{d}t, \tag{2}$$

where N_n represents the number of CPU and CPU(t) represents the usage rate of the CPU at the t moment. When the consumption is time, each transaction generated consumes a corresponding amount of time, written as follows:

$$T = Nt, \tag{3}$$

where T is total time, N is the number of transaction, and t is the time of per transaction. When the consumption is human resources, depending on the educational application, human resources will change as well. The larger an application project, the more manpower is required, and the more costly it is for different manpower to perform their respective roles in the application. When the consumption is memory resource, every time a block is added, the corresponding memory resource is consumed, written as follows:

$$M = Bb, \tag{4}$$

where M is the cost of memory resource, B is the memory size of block, b is the number of new block added. Compared with the traditional data sharing and certificate certification costs, after using the blockchain technology, it is not necessary to separate the ultra-large capacity central database for data storage, do not need to spend a lot of paper resources for certificate issuance, and do not need to hire third-party irrelevant personnel for supervision and management, which greatly reduces the cost of manpower and material resources.

4.3. Throughput. It refers to the number of transactions made per unit time for blocks with transactions on the blockchain, and the data interaction within blocks without transactions. Block transaction data can be used to gauge the system's throughput during the real application process. For instance, in a blockchain-based credit transfer framework, the faster the system responds to an application confirmation when multiple users submit credit transfer requests, the better the system throughput. Similarly, in a data-sharing framework, the faster the system throughput is, the more users are permitted to access shared resources.

4.4. Efficiency. It refers to the number of transactions that can be processed per unit of resource on the blockchain. This unit resource can be a memory resource, it can be a CPU resource, also can be a time resource, assuming that the efficiency is P, the number of nodes in the blockchain is N, the node ID is i, and different resource types are calculated differently. When the unit resource is a memory resource, the following expression is obtained:

$$P = \frac{T_s}{\sum_{i=1}^{N} \int_{ts}^{te} (Ai(t) + Bi(t))\mathrm{d}t}, \tag{5}$$

where $A_i(t)$ and $B_i(t)$ are represented as the occupied memory and running memory of node i at t moment, respectively. When the unit resource is a CPU resource, the following expression is obtained:

$$P = \frac{Ts}{\sum_{i=1}^{N} Nn \bullet \int_{ts}^{te} CPU_i(t)\mathrm{d}t}. \tag{6}$$

When the unit resource is a time resource:

$$P = \frac{Ts}{S}, \tag{7}$$

where S represents the total number of blocks.

4.5. Reliability. Reliability mostly pertains to the maximum number of malicious nodes that can exist, as there will unavoidably be malicious nodes on the blockchain. The ratio of the maximum number of malicious nodes to the total number of nodes can be accommodated when the blockchain is functioning smoothly. General data storage systems and data sharing platforms will fall collectively as a result of security threats, exposing the user's data. Following the development of the blockchain technology, every block in the chain is now interdependent and mutually contained, and data are saved using encryption techniques. If the node is attacked by a malicious party, this node is invalid, and the data of all other nodes are not impacted in any way.

4.6. Scalability. It refers to storage capacity and usage scenarios. Due to the restricted storage capacity of paper certificates and the use of a single storage technique, the record information that can be saved will be significantly constrained, raising doubts about the validity of the certificate. After the blockchain is combined, information can be saved through block nodes, a variety of smart contracts can be introduced; students, schools, governments, and employers can be included in the contract, multiparty authentication can be improved, and the credibility of data information can be increased. The ensuing contract can also be modified in accordance with various educational scenarios, and the scalability is relatively high.

4.7. Consistency. It refers to information that does not change over time. To prevent tampering with the data saved in the blockchain, decentralized distributed storage is used. The longer block data are kept unmodified, the higher the degree of authentication of the accessed data, the more robust the consistency. For instance, in the blockchain-based higher education transcript storage system proposed by Arndt and Guercio [105], the grade information of educated peoples after receiving higher education is stored in the system, and after a few years, the information queried by authorized users will not change. The more stable the system is, the better the performance will be.

4.8. Maintainability. It refers to problems that arise later and consume low cost. The blockchain is in a peer-to-peer network, there is no third-party control, and the blocks are equal nodes, avoiding a single point of failure and effectively lowering the risk of the system failing as a whole due to a small attack. This is in contrast to the centralized network of the traditional education system. The credibility and usefulness of a blockchain-based education system increase once it is put into operation because nodes watch out for one another, lowering error rates and maintenance costs down the road.

4.9. Real Time. It refers to calculate the time required to collect the data. The quicker the time, the more accurate the data are proven to be. The block of stored data will be stamped with a time stamp after it is added to the blockchain; this ensures that the data cannot be tampered with. The block of stored data will then be encrypted and saved. For instance, in the use of the graduate diploma storage system suggested by Schr and Mösli [106], when the educated people's graduation information is kept on the blockchain, the record is permanently maintained and no alteration is allowed, avoiding the issue of certificate fraud.

4.10. Processability. In the education process, many intermediate data are generated, which must all be recorded and saved. The block timestamp is verified for subsequent authentication and traces the authenticity of the data source. The blockchain technology can record users' learning data of formal education, learning data of nonformal education, cross-platform learning experiences and learning outcomes in a timely manner, and record index values are encrypted in a public key manner.

The abovementioned evaluation metrics are reflected in the construction of a credible system in educational application scenarios. The basic indicators are the characteristics

that every educational application system must have, and the basic performance of the system is reflected by the final results of these evaluation indicators. The evaluation of characteristic indicators is also involved in the existing applications. For example, scalability and consistency are reflected in [54, 76], process and real-time are reflected in [59, 74], and maintainability is reflected in [52, 57]. These evaluation metrics side-by-sides reflect that the blockchain technology plays a necessary role in building credible educational applications.

5. Conclusions

Establishing credibility in education is an urgent need to solve the optimization problem in education field at present. In this process, the blockchain technology shows its unique performance attributes, demonstrating its necessity for establishing credible applications in the education field. This paper focuses on the use of decentralized storage, privacy protection, and secure authentication of the blockchain technology to enhance trustworthiness and gives performance evaluation criteria for blockchain educational applications. They improve the privacy of education data storage, enhance the traceability of education data sharing, maintain the authenticity of education result authentication, ensure the fairness of education activity evaluation, and help build a credible system in the education field. However, considering the special nature of the education field itself, which involves many factors such as education environment, teaching methods, learning outcomes, and evaluation standards, the establishment of a credible system of the blockchain technology in the education field will face many challenges, such as limited storage space for education data, difficulties in authentication of education resources, and security issues of the blockchain technology itself.

5.1. Limited Storage Space. Rapid growth of education data, the limited storage space make it difficult to store all the education process data, which increases the difficulty of data traceability, reduces the authenticity of data sources and weakens the credibility of results. With the mature application of big data technology in the field of education, the data generated in educational activities has jetted up so much that the blocks in the blockchain need to carry more and more data, and the demand for storage space has become higher and higher. The data volume of educator information, educated person information, learning records, and certificate information is getting larger and larger, which will lead to serious limitation of data storage space, affecting the speed of storage and update of education data information, and also reducing the efficiency of user access to data. It is suggested to combine cloud storage, put a large amount of user information on the cloud, and store index values on the blockchain, which improves storage efficiency, ensures data authenticity and security, and also reduces the storage pressure of the blockchain.

5.2. Ambiguous Resource Rights. Data property rights are disputed, educational data are virtual, the authenticity of data sources cannot be confirmed nor can they be used as a basis for evaluation of educational activities, and the validity of evaluation results can be questioned. Compared with the real existence of data in the real world, the virtual nature of data on blockchain networks has become problematic. The relevant authorities should formulate corresponding regulations to clarify the ownership of data in order to confirm the rights of virtual data on the blockchain network. The data producer owns all the rights to the data, and any user who wants to use the data must get permission from the producer and provide something of value in exchange. The results obtained by the user through data analysis should be reasonably shared with the data producer by reaching a corresponding agreement.

5.3. Issue of Blockchain. The security of the blockchain itself still needs to be strengthened, and the construction of a credible system for educational application scenarios will also face security challenges. The development and use of blockchain's anonymity protection technology are not mature enough, and the management of keys is still at an early stage of development. With the development of cryptography and other technologies, whether the key will be cracked and whether it will cause information leakage afterwards, leading to a crisis of trust in education data. Consider multiple collaborative protection of the blockchain in time, space, technology, and other dimensions to better maintain the security of the blockchain.

While there are many issues that need to be explored in depth, this research opens a window of opportunity to better address trust relationships in education. As blockchain continues to develop, future applications in education will become more widespread and deeper, taking a research step in the direction of building a trustworthy system for education exploratory step.

Data Availability

The data supporting this review are from previously reported studies and datasets, which have been cited.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the Key Program of Chongqing Education Science Planning Project (K22YE205098) and the Doctoral Research Foundation of Chongqing Normal University (Nos. 21XLB030, and 21XLB029).

References

[1] J. Fan and X. Li, "Educational trust: the key to enhance the credibility of ideological and political education," *Journal of Henan Normal University (Natural Science)*, vol. 49, pp. 144–150, 2022.

[2] A. S. Anwar, U. Rahardja, A. G. Prawiyogi, N. P. L. Santoso, and S. Maulana, "iLearning model approach in creating blockchain based higher education trust," *International Journal of Artificial Intelligence*, vol. 6, no. 1, 2021.

[3] Q. Li and X. Zhang, "Blockchain:A technology to win open and trust in education," *The Journal of Distance Education*, vol. 35, no. 1, pp. 36–44, 2017.

[4] A. A. Monrat, O. Schelén, and K. Andersson, "A survey of blockchain from the perspectives of applications, challenges, and opportunities," *IEEE Access*, vol. 7, Article ID 117134, 117151 pages, 2019.

[5] L. Ouyang, "Smart contracts: architecture and research progresses," *Acta Automatica Sinica*, vol. 45, no. 3, pp. 445–457, 2019.

[6] S. Nithya and E. G. D. P. Raj, "Survey on asymmetric key cryptography algorithms," *Journal of Advanced Computing and Communication Technology*, vol. 2, no. 1, pp. 1–4, 2014.

[7] W. Ren, J. Hu, T. Zhu, Y. Ren, and K. K. R. Choo, "A flexible method to defend against computationally resourceful miners in blockchain proof of work," *Information Sciences*, vol. 507, pp. 161–171, 2020.

[8] Q. Zhang, "Incentive mechanism for federated learning based on blockchain and Bayesian game," *Scientia Sinica*, vol. 52, no. 6, pp. 971–991, 2022.

[9] W. Li, M. He, and H. Sang, "An overview of blockchain technology: applications, challenges and future trends," in *Proceedings of the 2021 IEEE 11th International Conference on Electronics Information and Emergency Communication (ICEIEC)*, Beijing, China, June 2021.

[10] A. Firdaus, M. F. A. Razak, A. Feizollah, I. A. T. Hashem, M. Hazim, and N. B. Anuar, "The rise of "blockchain": bibliometric analysis of blockchain study," *Scientometrics*, vol. 120, pp. 1289–1331, 2019.

[11] A. Hasselgren, K. Kralevska, D. Gligoroski, S. A. Pedersen, and A. Faxvaag, "Blockchain in healthcare and health sciences—a scoping review," *International Journal of Medical Informatics*, vol. 134, Article ID 104040, 2020.

[12] D. Li, D. Han, Z. Zheng et al., "MOOCsChain: a blockchain-based secure storage and sharing scheme for MOOCs learning," *Computer Standards & Interfaces*, vol. 81, no. 2022, Article ID 103597, 2022.

[13] T. Wang, S. Chang Liew, and S. Zhang, "Pubchain: a decentralized open-access publication platform with participants incentivized by blockchain technology," in *Proceedings of the 2020 International Symposium on Networks, Computers and Communications (ISNCC)*, IEEE, Montreal, QC, Canada, October 2020.

[14] S. Kwak and J. Lee, "Implementation of blockchain based P2P energy trading platform," in *Proceedings of the 2021 International Conference on Information Networking (ICOIN)*, IEEE, Jeju Island, South Korea, January 2021.

[15] H. Xu, "Trusted sharing platform of online education resources based on blockchain," *Wireless Internet Technology*, vol. 19, no. 13, pp. 63–65, 2022.

[16] S. Ølnes, J. Ubacht, M. Janssen, and M. Janssen, "Blockchain in government: Benefits and implications of distributed ledger technology for information sharing," *Government Information Quarterly*, vol. 34, no. 3, pp. 355–364, 2017.

[17] Z. Meng and Y. Wang, "Asymmetric encryption algorithms: primitives and applications," in *Proceedings of the 2022 IEEE 2nd International Conference on Electronic Technology, Communication and Information (ICETCI)*, IEEE, Changchun, China, May 2022.

[18] A. Anjum, M. Sporny, and A. Sill, "Blockchain standards for compliance and trust," *IEEE Cloud Computing*, vol. 4, no. 4, pp. 84–90, 2017.

[19] V. Buterin, "A next-generation smart contract and decentralized application platform," *white paper*, vol. 3, no. 37, pp. 2–1, 2014.

[20] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," *Decentralized Business Review*, Article ID 21260, 2008.

[21] R. Sujeetha and C. A. S. Deiva Preetha, "A literature survey on smart contract testing and analysis for smart contract based blockchain application development," in *Proceedings of the 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC)*, IEEE, Trichy, India, October 2021.

[22] J. Jayabalan and N. Jeyanthi, "A study on distributed consensus protocols and algorithms: the backbone of blockchain networks," in *Proceedings of the 2021 International Conference on Computer Communication and Informatics (ICCCI)*, IEEE, Coimbatore, India, January 2021.

[23] D. Puthal, N. Malik, S. P. Mohanty, E. Kougianos, and G. Das, "Everything you wanted to know about the blockchain: its promise, components, processes, and problems," *IEEE Consumer Electronics Magazine*, vol. 7, pp. 6–14, 2018.

[24] M. Dotan, Y. A. Pignolet, S. Schmid, S. Tochner, and A. Zohar, "Survey on blockchain networking: context, state-of-the-art, challenges," *ACM Computing Surveys*, vol. 54, no. 5, pp. 1–34, 2021.

[25] J. Pan, Z. Song, and H. Wangze, "Development in consensus protocols: from PoW to PoS to DPoS," in *Proceedings of the 2021 2nd International Conference on Computer Communication and Network Security (CCNS)*, IEEE, Xining, China, July 2021.

[26] T. P. Keenan, "Alice in blockchains: surprising security pitfalls in PoW and PoS blockchain systems," in *Proceedings of the 2017 15th Annual Conference on Privacy, Security and Trust (PST)*, IEEE, Calgary, AB, Canada, August 2017.

[27] N. Lasla, L. Al-Sahan, M. Abdallah, and M. Younis, "Green-PoW: an energy-efficient blockchain proof-of-work consensus algorithm," *Computer Networks*, vol. 214, Article ID 109118, 2022.

[28] M. Alzayat, J. Messias, B. Chandrasekaran, K. P. Gummadi, and P. Loiseau, "Modeling coordinated vs. P2P mining: an analysis of inefficiency and inequality in proof-of-work blockchains," 2021, https://arxiv.org/abs/2106.02970.

[29] S. . E. Thomsen and B. Spitters, "Formalizing nakamoto-style proof of stake," in *Proceedings of the 2021 IEEE 34th Computer Security Foundations Symposium (CSF)*, IEEE, Dubrovnik, Croatia, June 2021.

[30] K. Chen, "A formal analysis method of PoS consensus protocol based on byzantine fault tolerance," *Netinfo Security*, vol. 21, no. 8, pp. 35–42, 2021.

[31] J. Neu, S. Sridhar, L. Yang, D. Tse, and M. Alizadeh, "Securing proof-of-stake nakamoto consensus under bandwidth constraint," 2021, https://arxiv.org/abs/2111.12332.

[32] H. Xu, "Consensus protocol based on DPOS and aggregate signature," in *Proceedings of the 2022 3rd International Conference on Computer Vision, Image and Deep Learning & International Conference on Computer Engineering and*

Applications (CVIDL & ICCEA), IEEE, Changchun, China, May 2022.

[33] A. C. An, P. T. X. Diem, L. T. T. Lan, T. V. Toi, and L. D. Binh, "Building a product origins tracking system based on blockchain and PoA consensus protocol," in *Proceedings of the 2019 International Conference on Advanced Computing and Applications (ACOMP)*, IEEE, Nha Trang, Vietnam, November 2019.

[34] A. Ahmad, "Performance evaluation of consensus protocols in blockchain-based audit systems," in *Proceedings of the 2021 International Conference on Information Networking (ICOIN)*, IEEE, Jeju Island, Korea, January 2021.

[35] R. Ezzine, "A rigorous proof of the capacity of MIMO gauss-Markov Rayleigh fading channels," in *Proceedings of the 2022 IEEE International Symposium on Information Theory (ISIT)*, IEEE, Espoo, Finland, June 2022.

[36] X. Zhu, Y. Li, L. Fang, and P. Chen, "An improved proof-of-trust consensus algorithm for credible crowdsourcing blockchain services," *IEEE Access*, vol. 8, Article ID 102177, 102187 pages, 2020.

[37] J. Zhang, R. Tian, Y. Cao et al., "A hybrid model for central bank digital currency based on blockchain," *IEEE Access*, vol. 9, Article ID 53589, 53601 pages, 2021.

[38] A. Pal and K. Kant, "DC-PoET: proof-of-elapsed-time consensus with distributed coordination for blockchain networks," in *Proceedings of the 2021 IFIP Networking Conference (IFIP Networking)*, IEEE, Espoo, Finland, June 2021.

[39] D. Huang, X. Ma, and S. Zhang, "Performance analysis of the raft consensus algorithm for private blockchains," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 50, no. 1, pp. 172–181, 2020.

[40] A. S. Sharifovich, H. X. Maxmudovich, and B. M. Mansurovich, "Protocol for electronic digital signature of asymmetric encryption algorithm, based on asymmetric encryption algorithm based on the complexity of prime decomposition of a sufficiently large natural number," *Texas Journal of Multidisciplinary Studies*, vol. 7, pp. 238–241, 2022.

[41] C. Sullivan and E. Burger, "E-residency and blockchain," *Computer Law & Security Report*, vol. 33, pp. 470–481, 2017.

[42] G. Verma, M. Liao, D. Lu, W. He, X. Peng, and A. Sinha, "An optical asymmetric encryption scheme with biometric keys," *Optics and Lasers in Engineering*, vol. 116, pp. 32–40, 2019.

[43] X. Zhou and X. Tang, "Research and implementation of RSA algorithm for encryption and decryption," in *Proceedings of the 2011 6th international forum on strategic technology*, IEEE, Harbin, Heilongjiang, August 2011.

[44] Y. Song, "Research on security communication and application based on RSA algorithm," *Electronics Test*, vol. 16, pp. 33–36, 2021.

[45] L. X. Van and D. Hong, "Constructing a digital signature algorithm based on the difficulty of some expanded root problems," in *Proceedings of the 2019 6th NAFOSTED Conference on Information and Computer Science (NICS)*, IEEE, Hanoi, Vietnam, December 2019.

[46] F. J. Aufa and A. Affandi, "Security system analysis in combination method: RSA encryption and digital signature algorithm," in *Proceedings of the 2018 4th International Conference on Science and Technology (ICST)*, IEEE, Yogyakarta, Indonesia, August 2018.

[47] F. Liu, "Research on random encryption scheme of RSA algorithm and ECC algorithm," *Journal of Fujian Computer*, vol. 37, no. 8, pp. 4–7, 2021.

[48] L. Gong, "Design of network information security encryption system based on improved ECC algorithm," *China Computer & Communication*, vol. 34, no. 3, pp. 227–229, 2022.

[49] A. Alarifi, M. Amoon, M. H. Aly, and W. El-Shafai, "Optical PTFT asymmetric cryptosystem-based secure and efficient cancelable biometric recognition system," *IEEE Access*, vol. 8, Article ID 221246, 221268 pages, 2020.

[50] C. Turcu, C. Turcu, and I. Chiuchisan, "Blockchain and its potential in education," 2019, https://arxiv.org/abs/1903.09300.

[51] A. Mikroyannidis, J. Domingue, M. Bachler, and K. Quick, "A learner-centred approach for lifelong learning powered by the blockchain," *EdMedia+ Innovate Learning*, Association for the Advancement of Computing in Education (AACE), Chesapeake, VA, USA, 2018.

[52] H. Shen and Y. Xiao, "Research on online quiz scheme based on double-layer consortium blockchain," in *Proceedings of the 2018 9th International Conference on Information Technology in Medicine and Education (ITME)*, IEEE, Hangzhou, China, October 2018.

[53] G. Zou, "Designing a credit bank model based on blockchain technology," *Scientific and Social Research*, vol. 4, pp. 42–49, 2022.

[54] A. Rajalakshmi, K. Lakshmy, M. Sindhu, and P. Amritha, "A blockchain and ipfs based framework for secure research record keeping," *International Journal of Pure and Applied Mathematics*, vol. 15, pp. 1437–1442, 2018.

[55] L. Liu and S. Li, "Investigating the Impact of Bank Housing Credit Risk Control Strategy by Blockchain Technology on the Household Consumption Plan," *Computational Intelligence and Neuroscience*, vol. 2022, Article ID 7021384, 12 pages, 2022.

[56] W. Liang, Y. Yang, C. Yang, Y. Hu, S. Xie, and K. C. Li, "PDPChain: A Consortium Blockchain-Based Privacy protection Scheme for Personal Data," *IEEE Transactions on Reliability*, pp. 1–13, 2022.

[57] J. Rooksby and K. Dimitrov, "Trustless education? A blockchain system for university grades," *Ubiquity: The Journal of Pervasive Media*, vol. 6, no. 1, pp. 83–88, 2019.

[58] S. Kosasi, U. Rahardja, N. Lutfiani, E. P. Harahap, and S. N. Sari, "Blockchain technology-emerging research themes opportunities in higher education," in *Proceedings of the 2022 International Conference on Science and Technology (ICOSTECH)*, IEEE, Batam City, Indonesia, February 2022.

[59] H. Al, F. A. Shuhaimi, and K. K. J. Al Ismaily, "The upcoming Blockchain adoption in Higher-education: requirements and process," in *Proceedings of the 2019 4th MEC International Conference on Big Data and Smart City (ICBDSC)*, IEEE, Muscat, Oman, Janauary 2019.

[60] D. Shah, D. Patel, J. Adesara, P. Hingu, and M. Shah, "Exploiting the capabilities of blockchain and machine learning in education," *Augmented Human Research*, vol. 6, no. 1, pp. 1–14, 2021.

[61] M. Mulyati, I. Ilamsyah, A. Aris, I. Gunawan, and M. Suzaki Zahran, "Blockchain technology: can data security change higher education much better?" *International Journal of Cyber and IT Service Management*, vol. 1, no. 1, pp. 121–135, 2021.

[62] M. Turkanović, M. Holbl, K. Kosic, M. Hericko, A. Kamisalic, and A. Kamišalić, "EduCTX: A EduCTX: A Blockchain-Based Higher Education Credit Platformlockchain-Based Higher Education Credit Platform," *IEEE access*, vol. 6, pp. 5112–5127, 2018.

[63] P. Ocheja, B. Flanagan, and H. Ogata, "Connecting decentralized learning records: a blockchain based learning analytics platform," in *Proceedings of the 8th International Conference on Learning Analytics and Knowledge*, Sydney, Australia, March 2018.

[64] B. Awaji, S. Ellis, and L. Marshall, "Investigating the requirements for building a blockchain-based achievement record system," in *Proceedings of the 5th International Conference on Information and Education Innovations*, London, United Kingdom, August 2020.

[65] H. Li and D. Han, "EduRSS: a blockchain-based educational records secure storage and sharing scheme," *IEEE Access*, vol. 7, Article ID 179273, 179289 pages, 2019.

[66] F. Liu, "Research on the educational resources sharing framework based on blockchain," *Modern Educational Technology*, vol. 28, no. 11, pp. 114–120, 2018.

[67] E. Daniel and F. Tschorsch, "IPFS and friends: a qualitative comparison of next generation peer-to-peer data networks," *IEEE Communications Surveys & Tutorials*, vol. 24, pp. 31–52, 2022.

[68] X. Wang, "Design of educational data sharing platform based on blockchain technology," *Industrial Technology Innovation*, vol. 8, no. 2, pp. 31–36, 2021.

[69] F. Gao, "Study on construction of high-quality education resources platform framework based on blockchain," *Plateau Science Research*, vol. 5, no. 2, pp. 117–124, 2021.

[70] F. P. Oganda, N. Lutfiani, Q. Aini, U. Rahardja, and A. Faturahman, "Blockchain education smart courses of massive online open course using business model canvas," in *Proceedings of the 2020 2nd International Conference on Cybernetics and Intelligent System (ICORIS)*, IEEE, Manado, Indonesia, October 2020.

[71] H. Nusantoro, P. A. Sunarya, N. P. L. Santoso, and S. Maulana, "Generation smart education learning process of blockchain-based in universities," *Blockchain Frontier Technology*, vol. 1, no. 1, pp. 21–34, 2021.

[72] P. Ocheja, B. Flanagan, H. Ogata, and S. S. Oyelere, "Visualization of education blockchain data: trends and challenges," *Interactive Learning Environments*, pp. 1–25, 2022.

[73] T. Alam, M. Benaida, and B. Mohamed, "Blockchain and internet of things in higher education," *Universal Journal of Educational Research*, vol. 8, no. 5, pp. 2164–2174, 2020.

[74] S. Gilda and M. Mehrotra, "Blockchain for student data privacy and consent," in *Proceedings of the 2018 International Conference on Computer Communication and Informatics (ICCCI)*, IEEE, Coimbatore, India, January 2018.

[75] G. Zhao, H. Hui, and D. Bingbing, "Design and implementation of the digital education resources authentication system based on blockchain," in *Proceedings of the 2020 4th International Conference on Cryptography, Security and Privacy*, Nanjing China, January 2020.

[76] C. H. Han, G. J. K. Han, O. A. Gwang-Jun Kim, A. T. Osama Alfarraj, and Y. R. Amr Tolba, "ZT-BDS: a secure blockchain-based zero-trust data storage scheme in 6G edge IoT," *Journal of Internet Technology*, vol. 23, pp. 289–295, 2022.

[77] A. A. Gde, H. Nugroho, and R. Hendriyanto, "A blockchain-based halal certificate recording and verification prototype," *JOIV: International Journal on Informatics Visualization*, vol. 6, no. 2, pp. 364–370, 2022.

[78] B. Wu and Y. Li, "Design of evaluation system for digital education operational skill competition based on blockchain," in *Proceedings of the 2018 IEEE 15th International Conference on E-Business Engineering (ICEBE)*, IEEE, Xi'an, China, October 2018.

[79] X. Chen, "Blockchain simulation: a web application for it education," in *Proceedings of the 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*, IEEE, Las Vegas, NV, USA, January 2021.

[80] R. Q. Castro and M. Au-Yong-Oliveira, "Blockchain and higher education diplomas," *European Journal of Investigation in Health, Psychology and Education*, vol. 11, pp. 154–167, 2021.

[81] Z. Tian, M. Li, M. Qiu, Y. Sun, and S. Su, "Block-DEF: a secure digital evidence framework using blockchain," *Information Sciences*, vol. 491, pp. 151–165, 2019.

[82] Q. Liu, Q. Guan, X. Yang, H. Zhu, G. Green, and S. Yin, "Education-industry cooperative system based on blockchain," in *Proceedings of the 2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN)*, IEEE, Shenzhen, China, August 2018.

[83] M. Sanni and D. Apriliasari, "Blockchain technology application: authentication system in digital education," *Aptisi Transactions on Technopreneurship (ATT)*, vol. 3, no. 2, pp. 151–163, 2021.

[84] M. Han, Z. Li, J. He, D. Wu, Y. Xie, and A. Baba, "A novel blockchain-based education records verification solution," in *Proceedings of the 19th Annual SIG Conference on Information Technology Education*, Fort Lauderdale, FL, USA, September 2018.

[85] R. Arenas and P. Fernandez, "CredenceLedger: a permissioned blockchain for verifiable academic credentials," in *Proceedings of the 2018 IEEE International Conference on Engineering, Technology and Innovation (ICE/ITMC)*, IEEE, Stuttgart, Germany, June 2018.

[86] B. Awaji and S. Ellis, "Design, implementation, and evaluation of blockchain-based trusted achievement record system for students in higher education," 2022, https://arxiv.org/abs/2204.12547.

[87] I. B. Bandara, F. Ioras, and M. P. Arraiza, "The emerging trend of blockchain for validating degree apprenticeship certification in cybersecurity education," in *INTED2018 Proceedings*, pp. 7677–7683, 2018.

[88] M. Alshahrani, N. Beloff, and M. White, "Revolutionising higher education by adopting Blockchain technology in the certification process," in *Proceedings of the 2020 International Conference on Innovation and Intelligence for Informatics, Computing and Technologies (3ICT)*, IEEE, Piscataway, NJ, USA, December 2020.

[89] J. Woo, R. Fatima, C. J. Kibert, R. E. Newman, Y. Tian, and R. S. Srinivasan, "Applying blockchain technology for building energy performance measurement, reporting, and verification (MRV) and the carbon credit market: a review of the literature," *Building and Environment*, vol. 205, Article ID 108199, 2021.

[90] Z. A. Shaikh, A. A. Khan, L. Baitenova et al., "Blockchain hyperledger with non-linear machine learning: a novel and secure educational accreditation registration and distributed ledger preservation architecture," *Applied Sciences*, vol. 12, no. 5, Article ID 2534, 2022.

[91] M. A. Kusuma, P. Sukarno, and A. Aulia, *Security System for Digital Land Certificate Based on Blockchain and QR Code Validation in Indonesia*, EasyChair, Indonesia, 2022.

[92] J. Gresch, B. Rodrigues, E. Scheid, S. S. Kanhere, and B. Stiller, "The proposal of a blockchain-based architecture for transparent certificate handling," in *Proceedings of the*

International Conference on Business Information Systems, Springer, Amsterdam, Switzerland, 2018.

[93] T.-T. Kuo, "The anatomy of a distributed predictive modeling framework: online learning, blockchain network, and consensus algorithm," *JAMIA Open*, vol. 3, no. 2, pp. 201–208, 2020.

[94] C. Li, J. Guo, G. Zhang, Y. Wang, Y. Sun, and R. Bie, "A blockchain system for E-learning assessment and certification," in *Proceedings of the 2019 IEEE International Conference on Smart Internet of Things (SmartIoT)*, IEEE, Tianjin, China, August 2019.

[95] D. Lizcano, J. A. Lara, B. White, and S. Aljawarneh, "Blockchain-based approach to create a model of trust in open and ubiquitous higher education," *Journal of Computing in Higher Education*, vol. 32, no. 1, pp. 109–134, 2020.

[96] S. Solomon, "Blockchain Technology and Gamification-Conditions and Opportunities for education," in *Proceedings of the 8th International Adult Education 2018-Transformation In the Era Of Digitization And Artificial Intelligence*, Prague, Castle, March 2019.

[97] R. Bucea-Manea-Ţoniş, O. M. D. Martins, R. Bucea-Manea-Ţoniş et al., "Blockchain technology enhances sustainable higher education," *Sustainability*, vol. 13, no. 22, Article ID 12347, 12347 pages, 2021.

[98] R. Widayanti, E. P. Harahap, N. Lutfiani, F. P. Oganda, and I. S. P. Manik, "The impact of blockchain technology in higher education quality improvement," *Jurnal Ilmiah Teknik Elektro Komputer dan Informatika*, vol. 7, no. 2, pp. 207–216, 2021.

[99] Y. Zheng, "Design of a blockchain-based e-portfolio evaluation system to assess the education and teaching process," *International Journal of Emerging Technologies in Learning (iJET)*, vol. 16, no. 5, pp. 261–280, 2021.

[100] W. Zhao, K. Liu, K. Ma, and K. Ma, "Design of student capability evaluation system merging blockchain technology," *Journal of Physics: Conference Series*, vol. 1168, no. 3, p. 032123, 2019.

[101] V. Stepanova and I. Erins, "Assessment of blockchain-based professional growth data processing model," in *Proceedings of the 2020 the 4th International Conference on Business and Information Management*, Rome Italy, August 2020.

[102] M. Jirgensons and J. Kapenieks, "Blockchain and the future of digital learning credential assessment and management," *Journal of Teacher Education for Sustainability*, vol. 20, pp. 145–156, 2018.

[103] J. Xia and W. Zhang, "An overview of blockchain-based educational application systems," *Heilongjiang Science*, vol. 13, pp. 39–42, 2022.

[104] X. Lingling, "Blockchain technology research and application," *Chinese Journal of Scientific Instrument*, vol. 41, pp. 43–53, 2020.

[105] T. Arndt and A. Guercio, "Blockchain-based transcripts for mobile higher-education," *International Journal of Information and Education Technology*, vol. 10, pp. 84–89, 2020.

[106] F. Schär and F. Mösli, "Blockchain diplomas: using smart contracts to secure academic credentials," *Journal of Higher Education Research*, vol. 41, no. 3, pp. 48–58, 2019.

Hindawi
Security and Communication Networks
Volume 2023, Article ID 2413532, 14 pages
https://doi.org/10.1155/2023/2413532

Research Article

VM-Studio: A Universal Crosschain Smart Contract Verification and Execution Scheme

Tianxu Han,[1] **Jian Mao**,[1] **Sipeng Xie**,[1] **Qiyuan Gao**,[1] **Qin Wang**,[2] **Pinge Zhang**,[1] **and Yijia Fang**[1]

[1]*School of Cyber Science and Technology, Beihang University, Beijing 100191, China*
[2]*CSIRO Data61, Sydney, Australia*

Correspondence should be addressed to Jian Mao; maojian@buaa.edu.cn

Received 11 August 2022; Revised 29 October 2022; Accepted 22 February 2023; Published 18 April 2023

Academic Editor: Robert H. Deng

Blockchain interoperability promotes value delivery, application expansion, and ecological compatibility across heterogeneous blockchain systems. However, the contract framework and virtual machine construction in these systems are significantly different, and crosschaining becomes a challenging issue for system universality and compatibility. Starting from this problem, in this study, we propose VM-Studio, a crosschain smart contract verification and execution scheme to migrate the virtual machines (VMs) from the origin blockchain to the target blockchain. In our scheme, the migrated VMs are loaded as independent components enclosed in containers. We also design a unified system schedule to enable VM-Studio to allocate transactions into different containers. Loaded with origin blockchain VMs, these containers can accordingly solve crosschain transaction execution and smart contract verification. We implement VM-Studio and evaluate the transaction execution performance in the origin environment with multiple blockchains and the container environment. Experiment results demonstrate that VM-Studio achieves broad universality without compromising the execution performance of original blockchain transactions.

1. Introduction

A major benefit provided by blockchain is the ability to connect isolated individuals in a direct peer-to-peer way. Since Nakamoto [1] proposed Bitcoin in 2008, blockchain technology has been developed for over a decade, and various blockchain systems emerge constantly innovating in many aspects, such as consensus security [2, 3], privacy protection [4, 5], smart contract computing [6, 7], and blockchain scalability [8, 9]. However, this has also led to the phenomenon that they have formed separate value systems. The information barrier across heterogeneous blockchain systems makes value circulation and information transmission a complex problem. Therefore, crosschain technology, also known as interoperability [10], is equally important as a fundamental property to establishing Web3 ecosystems.

Crosschain technology is mainly oriented towards two aspects of the issue between blockchains, one is value exchange and the other is data transfer [11]. In the early understanding of crosschain technology, people only cared about how their assets were transferred from the one chain to another [12]. With the advent of the era from blockchains are smart contracts, assets no longer exist in the form of pure native tokens in the blockchain transaction sheet and tokens defined and managed by smart contracts gradually occupy the mainstream market [13]. In addition, in various smart contract ecosystems, algorithms [14] and components with powerful functions and data precipitation abound, making many blockchains have unique crosschain value. In order to make the crosschain value and mean of blockchain widespread, for a specific blockchain, we call it original blockchain. In the crosschain scenario, we will consider all aspects: transfer the value in the transaction order, call the algorithm controlled by the smart contract, and even migrate the upper Dapp application.

We note that the core of the above crosschain services is to verify the correctness of transaction orders and smart

contracts on other blockchains on the chain. It is not very easy to achieve between heterogeneous blockchains with different consensus mechanisms, single constructs of transactions, underlying contracts, and data interfaces. The existing crosschain projects need to negotiate with the target chain, including agreement, consensus, a trusted middleman, side-chain return to the target chain confirmation, and other operations. These schemes have a common problem: they are limited by the contract of the target chain and the architecture of the underlying virtual machine. When the contract language of the target chain is changed or the virtual machine of the target chain is upgraded, the corresponding crosschain scheme is no longer available. Therefore, a universal and multichain compatible universal scheme is needed for crosschain operation.

Considering the following scenario, two blockchains, the original blockchain and the target blockchain, are denoted as C_O and C_T, respectively. The state data controlled by the contract on C_O are transported to C_T, which is used for the construction of crosschain transactions by C_T. This process can be easily achieved if C_T supports smart contracts and is compatible with C_O's virtual machine (VM). As a result, crosschain smart contract verification can be easily realized by inputting the relevant transaction order and smart contracts from C_T into C_T's VM. However, the differences in consensus mechanisms, contract construction, and virtual machine construction between two blockchains become a huge hindrance. It is not easy to achieve interoperability due to their heterogeneity. If the underlying logic code of C_T's VM is modified or the data structures on C_O are modified to be compatible with the C_T's VM, the problem can be greatly mitigated. Moreover, it is also difficult for C_T to access the crosschain system for other heterogeneous blockchains. It can be seen that the construction of crosschain verification and execution schemes for smart contracts requires rigorous requirements on VM construction for both parties.

To sum up, after the abovementioned analysis, we give the primary motivation of the proposed scheme here. We note that crosschain technology mainly involves three parts: the transfer of value in the transaction order, the crosschain invocation of smart contracts, and the transplantation of Dapp ecology. All of them are related to the underlying verification logic of the blockchain virtual machine. We were inspired by Nervos CKB Polyjuice [15], which ran an EVM instance using CKB VM to implement a blockchain running a native account model within a blockchain of UTXO models. We believe that if this part of executing transactions and verifying smart contracts of blockchain virtual machines is decoupled separately and encapsulated by accessible unified services, crosschain execution and verification of transactions and contracts on heterogeneous blockchains can be realised, and it has the characteristics of strong universality, is easy to deploy and start, and is easy to upgrade.

1.1. Our Solution. We propose VM-Studio, a universal crosschain smart contract verification and execution scheme. We address the interoperability problems across heterogeneous blockchains by transforming the verification and execution of smart contracts into virtual machine migrations. Supposing to verify the contract states of C_O on C_T, we load the virtual machine image of blockchain C_O into a bare container. In an execution environment of C_T, C_O's transaction order is imported into the container as an atomic task. Then, C_O's VM execution environment can be simulated to verify and execute relevant transaction orders and smart contracts. VM-Studio allows any blockchain system to import its VM image into an empty container of another VM-Studio component (heterogeneous blockchain node), making the execution environment a *container service + virtual machine image*. Then, VM-Studio can manage and invoke all types of containers through scheduling tools. Due to the characteristics of container service, our scheme has no additional requirements on the VM construction among heterogeneous blockchains, showing strong universality. Based on that, *our contributions* are summarised as follows:

(i) We propose VM-Studio, a universal crosschain smart contract verification and execution scheme. We give the concrete construction of VM-Studio and elaborate on the message execution processes and crosschain verification of smart contracts.

(ii) We analyze the advantages of the VM-Studio scheme, including its correctness, universality, and low overhead in the execution of origin transactions and smart contracts.

(iii) We conduct experiments on crosschain smart contracts' verification and execution performance overhead. We compare them with the running results of each origin blockchain. We conclude that VM-Studio has satisfactory performance outcomes.

1.2. Our Advantages. Compared to more common crosschain tools, our solution has the following advantages:

(i) VM-Studio is compatible with a wide range of heterogeneous blockchains. Compared with the more traditional crosschain schemes, such as [12, 16], our scheme faces all heterogeneous blockchains that support virtual machines and has no specific requirements on the single transaction structure and smart contract architecture of the blockchain itself, which is convenient for the unified transplantation of all kinds of blockchains.

(ii) The native blockchain environment is relatively secure, and its VM is easy to upgrade. Existing crosschain projects [17, 18] that use side chain, relay chain, or parallel chain architecture require additional data interfaces outside the chain for crosschain verification and face tricky data availability problems when the chain version is upgraded. Our solution will be native blockchain transactions and smart contracts executed in their encapsulated original environments to maximize the security of the transaction execution process.

(iii) The scheme has a simple structure and is easy to implement. Compared with some recent research

[19, 20], the blockchain is used as the crosslink mechanism, supplemented by a consensus mechanism to ensure security; alternatively, some solutions [19] achieve cross-chain interoperability by using a unified programming language to call the smart contracts on different chains; they have specific difficulties in the implementation; that is, the construction of crosschain environments puts forward higher requirements. Our solution decouples the functions of blockchain virtual machines and wraps the templates for verifying transactions and smart contracts separately into containers. In the simplest case, our crosschain verification can be implemented in the local environment of nodes.

1.3. Paper Structure. Section 1 describes the scientific problems, solutions, contributions, and advantages proposed in this study. Section 2 provides related work. Section 3 defines the parameters and assumptions. Section 4 introduces the main construction of the VM-Studio scheme and presents the detailed workflow of four types of VM-Studio messages. Section 5 gives our experimental design and corresponding results. Section 6 discusses the performance and application of VM-Studio from different perspectives. Last, Section 7 concludes this study.

2. Related Work

2.1. Smart Contract. A smart contract is essentially a piece of the program stored on the blockchain that can be automatically executed according to specified contract rules. The concept of a smart contract was first proposed by Szabo [21] in 1994, aiming to build an efficient contract without ambiguity that could be enforced with the help of code. Then, smart contracts were widely adopted with the advent of Ethereum [22]. In Ethereum, smart contracts are executed inside the Ethereum Virtual Machine (EVM) and are isolated from external environments. To ensure the reasonable usage of resources, Ethereum adopts the gas billing standard, where any computing operation needs to pay a certain amount of gas as the cost, including the *creation*, *call*, *data acquisition,* and other contract operations. EoS [23] claims to propose a set of smart contract systems that are easier to develop, which solves the problems of low performance and high fees of Ethereum smart contracts. Hyperledger Fabric [24] is an open license blockchain designed for crossindustry development among enterprises. It is built on Linux with good modularity and structural characteristics [24]. It provides reliable services for many industries, as well as high portability and versatility. Chaincode, the smart contract on Hyperledger Fabric, runs in a Docker container. It is a program that can query or change the ledger's state. The final execution results are synchronized to every node in the network. However, due to the alliance chain's centralization concerns, Hyperledger Fabric is unsuitable for the public chain.

2.2. Blockchain Interoperability. Blockchain interoperability technologies focus on the verifiable transfer of on-chain tokens and states across different chains [11]. Typically, three types of methods have been adopted: *hash-time lock*, *third-party crosschain,* and *sidechains/relays* mechanisms.

2.2.1. Hash-Time Lock. The hash-time lock mechanism [16] adopts two core techniques: *hash lock* and *time lock*. Two parties first need to establish a communication channel. Then, the two locking techniques are used to lock their on-chain assets, and the negotiated hash value is used to unlock the on-chain holdings of the other party for crosschain exchanges. The hash-time lock has apparent disadvantages: both parties have to hold the corresponding accounts on two chains with sufficient assets, and both blockchains must support smart contracts for contract executions.

2.2.2. Third-Party Crosschain. The third-party crosschain mechanism, such as the notary mechanism [25], introduces a trusted individual or group as the notary by tracking the status of two chains, collecting evidence, and verifying the transaction to facilitate the crosschain trade. It can support the interoperability of heterogeneous blockchains. However, the solution can only support the crosschain functions for origin token exchanges, and both sides connected by the crosschain transactions have to bear significant centralization risks. Hash-time lock and third-party mechanisms are subject to architectural and security constraints [26].

2.2.3. Sidechains/Relays. The sidechains/relays crosschain mechanism [12, 17, 18, 27] solves the centralization risk. Assets on the main chain and sidechains are mapped one by one through the two-way anchoring technology. The lock and release mechanism realizes the bidirectional workflow between the main chain and side chains. The solution has advantages: it supports the invocation of a crosschain smart contract, which is no longer limited to simple token exchange [28]. However, it suffers the drawback of high costs because its adoption is limited. Among sidechains/relays schemes, the best known are Cosmos [17] and Polkadot [18]. Polkadot is a crosschain platform that combines heterogeneous chains. It realizes the network-wide diffusion of user anonymity and formal verification through parachains. In Polkadot, transactions on each parachain can be passed to other chains through the bridge, and these transactions can be executed and verified by multiple chains. However, Polkadot has problems such as difficulties in selecting a validator, governance, forming parachains, or intransparency of validator election [29]. Cosmos aims to build an interconnected blockchain ecosystem. It provides a relatively complete and convenient way of facilitating crosschain interaction through the *Tindermint* consensus engine, the *Cosmos SDK* modular development framework, and the interblockchain communication protocol. However, the system is relatively complex, which is difficult for the developers to understand, due to factors, such as its economic model and market competition, Cosmos suffers the pressure from nonpure technical reasons.

In addition, several studies on heterogeneous blockchain crosschain also deserve attention [19, 20]. HyperService [19] is a heterogeneous crosschain framework that programmers can freely develop. It provides a unified Dapp programming language at the bottom of the native blockchain and supports crosschain communication with smart contract construction. However, the development framework proposed by HyperService is relatively complex, such as the specific format of data, dynamical checks for accuracy, and data input interface requirements, which lack universality. The scheme proposed in this study has a lower migration cost than HyperService in terms of the new blockchain virtual machine because the former only requires simple VM image encapsulation and loading, while the latter needs to develop the underlying contract logic of the new blockchain into a unified interface, according to the relevant developer documents. When the smart contract or virtual machine version of the native blockchain needs to be upgraded, the HyperService must be redeveloped according to the new rules. Our solution only needs to call a virtual machine image of the latest version. Blockchain Router [20] establishes a set of crosslink routing rules for multiple blockchains to establish the routing path for crosschain communication. The process of crosschain data verification is completed by the nodes of the block link, which is different from the focus of this study.

3. Parameters and Assumptions

We denote a blockchain system by C. For multiple blockchain systems, use $C_1, \cdots, C_i, \cdots$, where i is a variable positive integer. For a blockchain system C_i, we make the following assumptions.

3.1. Chain Identifier. Every blockchain system, or chain, has a unique identity. The genesis block is identified with a string of hash value GenesisHash. For the blockchain C_i, we define its identifier id_{C_i} as follows:

$$id_{C_i} = \mathrm{H}_{C_i}\left(\mathrm{GenesisHash}_{C_i}\right). \tag{1}$$

Here, H_{C_i} is a hash function, where a random string of length $\{0,1\}^*$ is mapped to the range of id_{C_i}'s values. $\mathrm{GenesisHash}_{C_i}$ represents the hash value of the genesis block on C_i.

3.2. Chain State. By default, the state model of blockchain can be applied to both the *UTXO* and the *account*-based blockchains. For generality, we define the state model as follows.

Definition 1 (chain state). For the blockchain C_i, we define, when it is based on the account model, we denote the set of states corresponding to all accounts as the chain state of C_i; when it is based on the UTXO model, the set of states corresponding to all UTXOs is called the chain state of C_i.

It is worth noting that for the blockchain system with the account model, the chain state can be equivalent to the corresponding state root in the block header of the latest block, such as the concept of the world-state MPT root in Ethereum. For the blockchain system with the UTXO model, the chain state represents the set of outputs of all transactions for which the input from subsequent transactions has not been referenced: the set of all existing UTXOs on the blockchain. In either case, we abstract the chain state as a snapshot, which can reflect the latest state or data set of the blockchain in real time. Equivalently, there is a more straightforward method: execute all transactions on the blockchain in order, from the genesis block to the latest block. It can be seen that their essences are very similar.

Therefore, in this study, we mainly express the perspective of the account-based blockchain to have a more intuitive understanding. As described above, by specifying the chain state as a pointer field in the block header of the latest block, we can calculate and verify the states through the batch acquisition of transaction data. If all the transactions are known, the state database of the corresponding data structure can be calculated by existing rules. Then, the chain state pointer can be extracted by the corresponding rules. The correctness of the obtained block data can be verified by comparison. With the continuous generation of new caches, we can verify the correctness of new transaction orders by calling the cached snapshot. In addition, to facilitate the construction of invocation instances for crosschain smart contracts, we define a multichain system:

Definition 2 (multichain). A multichain system $M(n)$ is a multichain system consisting of a series of parallel blockchains $M = \{C_1, ..., C_n\}\,(n \geq 2)$ if for any $i \in [1, n]$, and for any $j \in [1, n]$, there is $id_{C_i} \neq id_{C_j}$.

The above feature ensures that a multichain system $M(n)$ does not contain two identical blockchains, even though their genesis blocks are consistent. Equivalently, when the Genesis blocks of two blockchains are the same, we allow the sets of states to be different, indicating that the two blockchains are branched from the same chain. Nevertheless, we do not care about this situation.

In addition, we introduce several concepts mentioned in the previous section. Consider the crosschain scenario of two blockchains, where we call the crosschain from the origin blockchain to the target blockchain, denoted by C_O and C_T, respectively.

4. Main Construction of VM-Studio

In this section, we begin with an overview of the main construction of VM-Studio, including its five main components and four message types. Later, in 4.2, we give the details of the four types of messages, the interaction process of the five main components of VM-Studio, the specific structure of the messages, and the verification process of the data.

4.1. Overview. The main construction of VM-Studio is shown in Figure 1. As a general chain state execution construction, VM-Studio components are attached to the

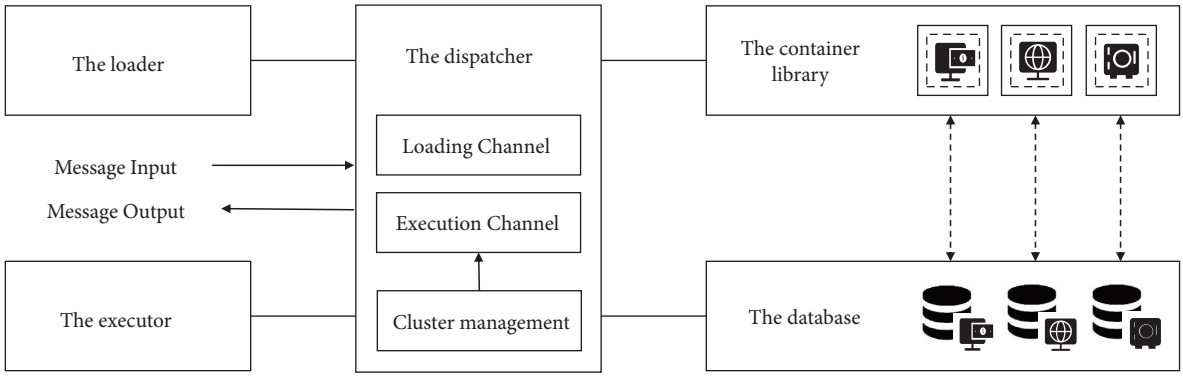


Figure 1: The structure of VM-Studio.

network nodes of a relay blockchain. The following sections will present concrete examples of running crosschain smart contracts under the intermediate blockchain. VM-Studio consists of five main modules: *the dispatcher*, which is responsible for overall dispatch, message input, and output; *the loader* is responsible for loading the VM image; *the executor*, which executes containers loaded with blockchain VMs; *the container library*, which stores the containers loaded with virtual machines; and *the database*, which is responsible for storing a snapshot of states as a database.

4.1.1. The Dispatcher. It is the core component of VM-Studio. It receives the input and output messages of message instructions externally and processes messages internally.

The dispatcher mainly receives four types of messages: *MessageI*, *Preload* mainly used to verify the correctness and integrity of packets and update the block synchronization height to start the subsequent formal loading process. *MessageII*, *load*, is mainly used to load the prepared virtual machine image into the empty container, transport it to the container library for storage, and open the access interface between it and the corresponding blockchain ledger in the database to start the subsequent execution process. *MessageIII*, *Execute* is mainly used to manage atomic transaction order messages through the cluster, and the container verifies transactions and executes smart contracts. *MessageIV*, *Preload and Update*, both the preload and virtual machine version update functions. The dispatcher first performs a preload process.

For the above four messages, the dispatcher has different message channels to process them: *MessageI*, *MessageII,* and *MessageIV* will be included in the loading channel; *MessageIII* is included in the execution channel and scheduled by the cluster management, an internal component of the dispatcher. This is because *MessageIII* is the atomic transaction order obtained after the block data are split, and the quantity is millions of times that of other types of messages. In order not to delay the processing of other types of messages but also to allocate the core resources for *MessageIII*, *MessageIII* is included in an independent message channel.

4.1.2. The Container Library. The container library is the container warehouse of VM-Studio, which is used to store containers after loading and presents the call interface with the blockchain *id* as the index. Cluster Management centrally arranges for the executor to call the container library.

4.1.3. The Database. The database is the back-end database of VM-Studio, which stores the blockchain ledger, state snapshot, state database, and virtual machine image. It presents the call interface with the blockchain *id* as the index. The VM can be directly called by the container where the VM of the corresponding blockchain resides.

4.1.4. The Loader. The loader is the loading center of VM-Studio. The loader is used to load virtual machine images into empty containers for *MessageII* services.

4.1.5. The Executor. It is the executor management center of VM-Studio. Triggered by *MessageIII*, the Cluster Management component of the dispatcher will send the call instruction to the executor for containers. The executor will then call the corresponding container in the container library through the call interface, that is, characterized by *id*.

Next, we will describe the workflow and interactions of VM-Studio after each type of message is given.

4.2. Details of the Scheme. In this section, we describe in detail how VM-Studio performs crosschain execution and verification of transactions and smart contracts when receiving the mentioned four types of messages, including those actively sent by nodes and those automatically generated by the dispatcher. The overall process is shown in Figure 2. Each type of message is a dataset packet dp consisting of a header dp_{head} and a body dp_{body}, while dp_{body} is mainly composed of block data.

4.2.1. Message Signature. First, we formally describe the user identity and signature of the node. In our setting, VM-Studio is positioned as a general blockchain crosschain data verification component; in a given blockchain system, a node that has installed and instantiated VM-Studio can verify

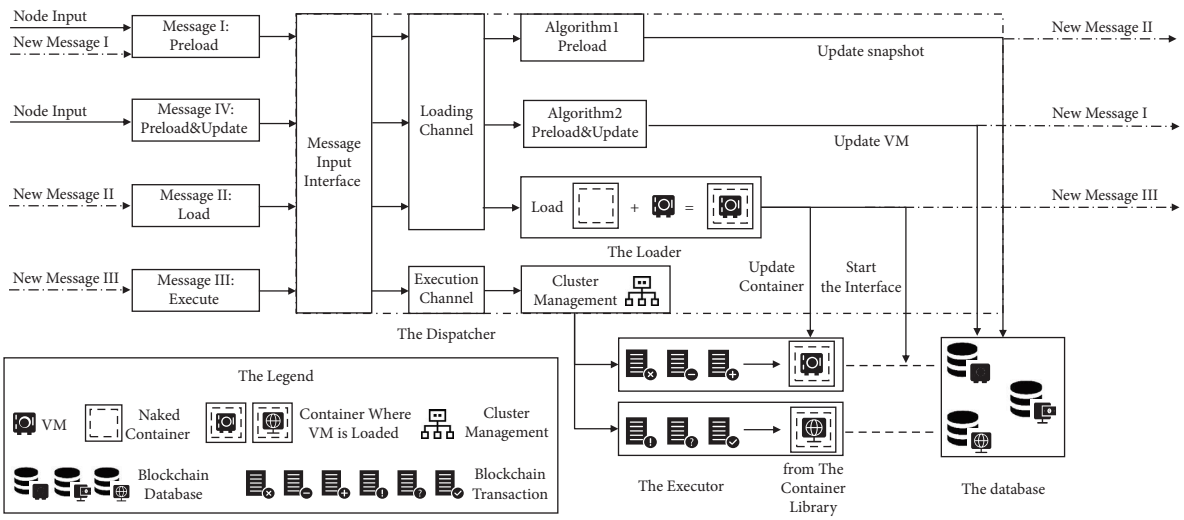


Figure 2: The operation flow of four kinds of messages.

crosschain data in the blockchain system. Considering that the identity of a node as a chain user is not fixed in different chain environments, we use a common representation of the public chain here. We have abstracted the more traditional blockchain mechanisms, such as Bitcoin and Ethereum. For a more detailed description, please refer to [1, 22]. For a node to sign the message M, the following three algorithms are formally defined, including the key generation algorithm KeyGen, the signature algorithm Sig, and the signature verification algorithm VerifySig.

$\mathrm{KeyGen}(\lambda, cp) \longrightarrow pk, sk, \mathrm{add\,r}$. λ is a security parameter, and cp is a blockchain parameter that contains information, such as the version byte of the blockchain used to distinguish between the main network and the test network. When a node joins the blockchain network, the private key sk is generated by a random number in a specific range and the public key pk is generated by sk in a trapdoor reversible way. Finally, the address $add\,r$ is generated by an irreversible algorithm: the address of the node's account in the blockchain network. It should be noted here that, first, the range of random numbers used to generate sk is determined by the security parameter λ. Second, we generally assume that the irreversible algorithm used is the hash function, H_{C_i}, and the calculation process is $add\,r = cp \| \mathrm{H}_{C_i}(pk)$.

$\mathrm{Sig}(M, sk) \longrightarrow sig_M$. sig_M is the result of signing the message M for a node using its own private key sk.

$\mathrm{VerifySig}(\mathrm{sig}_M, M, pk) \longrightarrow \{0, 1\}$. If the signature verification is successful, output is 1; otherwise, output is 0.

In general, in the practical application of blockchain, the message signature's object may be the message's hash value. In verifying the signature, it is necessary to verify that the related address $add\,r$ is from the specific chain version cp and matches the public key pk. In Algorithm 1, we present the verification procedure of the message signature when the input is $(sig, pk, \mathrm{add\,r}, cp, M^*)$.

4.2.2. MessageI, Preload. *MessageI* is initiated by the node or automatically initiated by the dispatcher after *MessageIV* is processed. In the previous article, we mentioned that *MessageI* is used to verify the correctness and integrity of data packets and update the block synchronization height to enable subsequent formal loading processes. When *MessageI* is correctly processed by the dispatcher, the dispatcher will automatically generate *MessageII* and send it to the input of the dispatcher's message channel. The specific process is as follows:

(i) *Step 1.* Input MessageI to the dispatcher.

(ii) *Step 2.* The dispatcher parses MessageI into

$$\left(\mathrm{type}, vpp_{\mathrm{sig}}, h_{\mathrm{start}}, h_{\mathrm{end}}, id\right). \tag{2}$$

The detailed structure of *MessageI* is listed in Table 1. According to $type$, the message type is *MessageI*.

(iii) *Step 3.* The dispatcher operates Algorithm 2, and accordingly outputs

$$\left(dp_{\mathrm{head}}, dp_{\mathrm{body}}, \mathrm{maxheight}, \mathrm{err}\right) \leftarrow \mathrm{PreLoad}\left(dp_{\mathrm{head}}, dp_{bpdy}\right). \tag{3}$$

If err equals to 1, break the process.

(iv) *Step 4.* The dispatcher parses dp_{head} to

$$\left(id, h_{\mathrm{start}}, h_{\mathrm{end}}, vpp_{\mathrm{sig}}\right), \tag{4}$$

and sends

$$\left(dp_{\mathrm{body}}, h_{\mathrm{start}}, h_{\mathrm{end}}, id\right), \tag{5}$$

as *MessageII* to the dispatcher's message channel.

4.2.3. MessageII, Load. *MessageII, Load* is automatically initiated by the dispatcher after MessageI is processed. Its main functions include loading the prepared virtual machine image into an empty container, transporting it to the container library for storage, and opening the access interface between it and the corresponding blockchain database for subsequent executions. When *MessageII* is correctly

Input: sig, pk, add r, cp, M^*
Output: b
Initialization: $1 \longleftarrow b$
If VerifySig(sig, $\mathrm{H}_{C_i}(M^*)$, pk)! = 1 **then**
  $0 \longleftarrow b$;
  return b;
end
If $add\,r! = cp \| \mathrm{H}_{C_i}(pk)$ **then**
  $0 \longleftarrow b$;
  return b;
end
return b

Algorithm 1: VerifyMessageSig.

Table 1: Message structure of *MessageI*.

Parameters	Meanings
type	Message type and it has the value I
vpp_{sig}	Publicly verifiable signature parameters, including parameters related to node identity: (i) sig: the node's signature for hash_{dp} (ii) pk: the node's public key (iii) add r: the node's address in blockchain network (iv) cp: the blockchain parameter that contains information such as the version byte of the blockchain used to distinguish between the main network and the test network (v) hash_{dp}: the hash value of the data package
h_{start}	The starting height of the synchronization block
h_{end}	The ending height of the synchronization block
id	The identifier of the blockchain from which the packet originated

Input: dp_{head}, dp_{body}
Output: dp_{head}, dp_{body}, maxheight, err
Initialization: $0 \longleftarrow b$, $0 \longleftarrow$ err, $0 \longleftarrow$ maxheight
parse $(id, h_{\mathrm{start}}, h_{\mathrm{end}}, vpp_{\mathrm{sig}}) \longleftarrow dp_{\mathrm{head}}$
parse $(\mathrm{sig}, pk, \mathrm{add\,r}, cp, \mathrm{hash}_{dp}) \longleftarrow vpp_{\mathrm{sig}}$
parse GenesisHash $\in dp_{\mathrm{body}}$
run $b \longleftarrow$ VerifyMessageSig(sig, pk, add r, cp, hash_{dp}), $b \in \{0, 1\}$
If $b == 0$ **then**
  $1 \longleftarrow$ err;
  return err;
end
If H(GenesisHash)! = id **then**
  $1 \longleftarrow$ err;
  return err;
end
query sn and maxheight for id
If $h_{\mathrm{start}} >$ maxheight or maxheight $\geq h_{\mathrm{end}}$ **then**
  $1 \longleftarrow$ err;
  return err;
else
  $h_{\mathrm{start}} \longleftarrow$ maxheight;
  maxheight $\longleftarrow h_{\mathrm{end}}$;
end
package $dp_{\mathrm{head}} \longleftarrow (id, h_{\mathrm{start}}, h_{\mathrm{end}}, vpp_{\mathrm{sig}})$
return dp_{head}, dp_{body}, maxheight, err

Algorithm 2: Preload.

processed by the dispatcher, the dispatcher will automatically generate *MessageIII* and send it to the input of the dispatcher's message channel. The main process is as follows:

(i) *Step 1*. Input *MessageII* to the dispatcher.

(ii) *Step 2*. The dispatcher parses *MessageII* to

$$\left(\text{type}, dp_{\text{body}}, h_{\text{start}}, h_{\text{end}}, id\right). \tag{6}$$

The detailed structure of *MessageII* is listed in Table 2. According to type, the message type is *MessageII*.

(iv) *Step 3*. The dispatcher loads the blockchain VM corresponding to blockchain *id* and inputs it to the loader. Meanwhile, the database interface corresponding to blockchain *id* is opened to the loader.

(v) *Step 4*. The loader uses the bare container to load the corresponding VM, which is expressed as Container $(VM(id))$.

(vi) *Step 5*. The loader links Container $(VM(id))$ to the interface of blockchain *id* in the database.

(vii) *Step 6*. The loader starts Container $(VM(id))$ and synchronizes blockchain *id* ledger from h_{start} to h_{end} to load data.

(viii) *Step 7*. After data are loaded, the dispatcher parses dp_{body} to atomic transactions and transmits each transaction to the input of the dispatcher's message channel in the form of *MessageIII*.

4.2.4. MessageIII, Execute. MessageIII, Execute is automatically initiated by the dispatcher after MessageII is processed. Its main use is to distribute atomic transaction in single messages through Cluster Management to the corresponding container, which then, guided by the executor, validates the transaction and executes the smart contract.

(i) *Step 1*. Input *MessageIII* to the dispatcher.

(ii) *Step 2*. The dispatcher parses *MessageIII* into

$$(\text{type}, id, tx). \tag{7}$$

The detailed structure of *MessageIII* is listed in Table 3. According to $type$, it is *MessageIII*. The dispatcher allocates this message to the execution channel.

(iii) *Step 3*. The executor passes the corresponding tx to the working Container $(VM(id))$ according to the *id* information in Message. If the container is not in working state, the executor fetches Container $(VM(id))$ from thecontainerlibrary according to *id*.

4.2.5. MessageIV, Preload and Update. MessageIV, Preload and Update is initiated by the node itself and can be regarded as an extended version of *MessageI*. It can preload and update virtual machine versions. The dispatcher will first perform a preload process and then determine whether the virtual machine of the corresponding blockchain needs to be updated. If so, the virtual machine image stored in the database will be updated. After the update, the dispatcher will automatically generate *MessageI* and moreover feed it to the input part of the dispatcher's message channel. The main process is as follows:

(i) *Step 1*. Input *MessageIV* to the dispatcher.

(ii) *Step 2*. The dispatcher parses *MessageI* into

$$\left(\text{type}, vpp_{\text{sig}}, h_{\text{start}}, h_{\text{end}}, id, VM - \text{version}\right). \tag{8}$$

The detailed structure of *MessageIV* is listed in Table 4. According to $type$, the message type is *MessageIV*.

(iii) *Step 3*. The dispatcher operates Algorithm 3 and then outputs

$$\left(dp_{\text{head}}, dp_{\text{body}}, \text{maxheight}, \text{isupdate}, \text{err}\right) \leftarrow \text{PreLoad\&Update}\left(dp_{\text{head}}, dp_{bpdy}\right). \tag{9}$$

If *err* equals to 1, break the process.

(iv) *Step 4*. If isupda te equals to 1, dispatcher parses dp_{head} into

$$\left(id, h_{\text{start}}, h_{\text{end}}, vpp_{\text{sig}}, VM - \text{version}\right). \tag{10}$$

The dispatcher updates the virtual machine corresponding to blockchain *id* of the database to the $VM - version$ version. The dispatcher packages

$$dp_{\text{head}} \leftarrow \left(id, h_{\text{start}}, h_{\text{end}}, vpp_{\text{sig}}\right), \tag{11}$$

and sends $(dp_{\text{head}}, dp_{\text{body}})$ as *MessageI* to the input of the dispatcher's message channel.

(v) *Step 5*. If isupda te does not equal to 1, dispatcher will parse

dp_{head} to $(id, h_{\text{start}}, h_{\text{end}}, vpp_{\text{sig}}, VM - \text{version})$ and sends

$$\left(dp_{\text{body}}, h_{\text{start}}, h_{\text{end}}, id\right), \tag{12}$$

as *MessageII* to the dispatcher's message channel.

5. Implementation and Experiment

In this section, we implement the VM-Studio prototype and report our experiment results by comparing our prototype with the original blockchain node implementations. We

Table 2: Message structure of *MessageII*.

Parameters	Meanings
type	Message type and it has the value *II*
dp_{body}	The body of the data packet, which contains the block data of the blockchain
h_{start}	The starting height of the synchronization block, included in the outputs of *MessageI*
h_{end}	The ending height of the synchronization block, included in the outputs of *MessageI*
id	The identifier of the blockchain from which the packet originated, which equals to that in *MessageI*

Table 3: Message structure of *MessageIII*.

Parameters	Meanings
type	Message type and it has the value *III*
id	The identifier of the blockchain from which the packet originated, which equals to that in *MessageII*
tx	An indivisible atomic transaction

Table 4: Message structure of *MessageIV*.

Parameters	Meanings
type	Message type and it has the value *IV*
vpp_{sig}	Publicly verifiable signature parameters, including parameters related to node identity: (i) sig: the node's signature for hash_{dp} (ii) pk: the node's public key (iii) add r: the node's address in blockchain network (iv) cp: the blockchain parameter that contains information such as the version byte of the blockchain used to distinguish between the main network and the test network (v) $hash_{dp}$: the hash value of the data package
h_{start}	The starting height of the synchronization block
h_{end}	The ending height of the synchronization block
id	The identifier of the blockchain from which the packet originated
VM − version	The VM version of the blockchain from which the packet originated

implemented our prototype in three unstable, dynamic environments and applied it to than 1,300,000 blocks with 1,000,000 transactions for testing.

5.1. Implementation Details. We use an elastic compute server with a 2.1 GHz Intel Xeon(R) gold 6,230 CPU, a 32 GB of 2,400 MHz DDR4 memory, a 1 TB solid state disc, and an Ubuntu 18.04 operating system to implement the proposed system. The node implementation (*Go-Ethereum*) was modified and could interact with the premade containers, which loaded virtual machines from each involved blockchain; we also employed the *go-metrics* package to add metrics to measure the execution time of the *contract call* function, the *contract create* function, and the RPC communication function. Full nodes are running on a proof-of-work blockchain named *Ropsten*, while operating on a proof-of-authority blockchain named *Görli* and the blockchain with a hybrid consensus engine named *BnB Smart Chain* (*BSC*, for short). For each blockchain, we test our prototype with more than 1,300,000 blocks and up to 1,000,000 transactions. Then, we compare the overhead of the origin VM with our implementation. The results are as follows.

5.2. Experiment Results. As described above, we run experiments on three blockchain networks: *Ropsten*, *Görli*, and *BnB Smart Chain*. To test out the performance of our prototype system, we measure the overhead of both the RPC communication and the virtual machine computation; after that, we conduct an overall overhead comparison between the VM-Studio and the origin system. The full node of each blockchain is deployed on the elastic compute server mentioned previously.

Firstly, we discuss the virtual machine computation overhead of VM-Studio. The three blockchain networks contain different transaction sets, resulting in different computation performances. As shown in Figure 3, the average computation overhead of the *Ropsten* blockchain goes up during the first 10,000 transactions and reaches the highest point of around 285 microseconds and then goes slightly down to about 150 microseconds until the total of 1,000,000 transactions are applied. Comparing the performance of our prototype system with the original system, the latency our system adds on is less than 35 microseconds, which may be a result of the data exchange between different functions inside the container. From our view, the added latency is acceptable for production usage.

Input: dp_{head}, dp_{body}
Output: dp_{head}, dp_{body}, $maxheight$, err
Initialization: $0 \leftarrow b$, $0 \leftarrow err$, $0 \leftarrow maxheight$ parse $(dp_{head}, VM - version) \leftarrow dp_{head}$
run Algorithm 2: $(dp_{head}, dp_{body}, naxheight, err) \leftarrow$ Preload (dp_{head}, dp_{body})
query VM version $version$ for id
If $VM - version > version$ **then**
  $1 \leftarrow isupdate$;
end
package $dp_{head} \leftarrow (id, h_{start}, h_{end}, vpp_{sig}, VM - version)$
return $dp_{head}, dp_{body}, maxheight, isupdate, err$

Algorithm 3: Preload and Update.

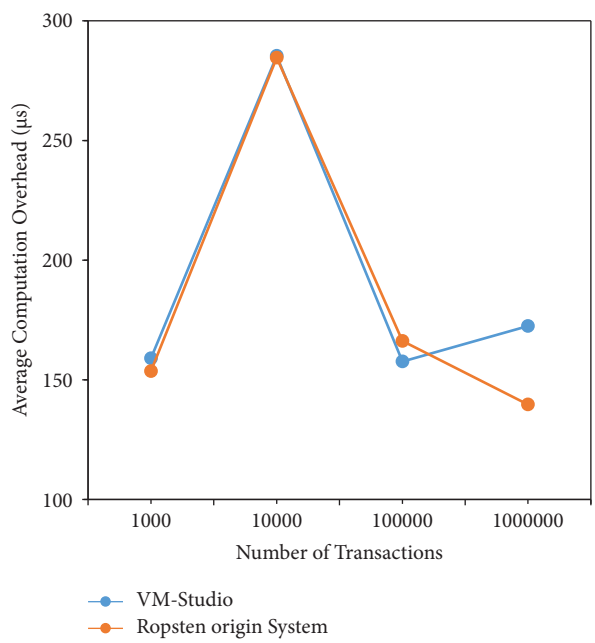


Figure 3: The average computation overhead on the Ropsten blockchain. The sampling size is fixed at 1,000,000 transactions, involving 402,361 blocks.

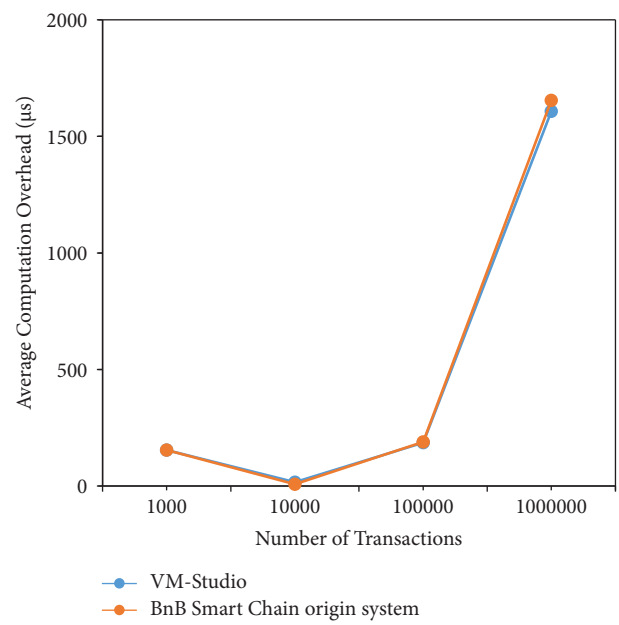


Figure 4: The average computation overhead on *BnB SmartChain*. The sampling size is fixed at 1,000,000 transactions, involving 1,311,838 blocks.

As for the *BnB smart contract* blockchain network, referring to Figure 4, the average computation overhead continues going down to about 10 microseconds within the first 10,000 transactions. Then, it keeps rising until it hits the summit of more than 1,600 microseconds when the virtual machine goes through the entire 1,000,000 transactions. The performance of both systems is relatively close, and our system performs slightly better when 1,000, 100,000, and 1,000,000 transactions come in. The reason is that transactions in the *BnB SmartChain* blockchain network are often related to more sophisticated smart contract codes. Therefore, our efficient EVM implementation presents better performance. VM-Studio offers about 8% performance improvement (computed by $(VO - OO)/OO$, where VO stands for VM-Studio overhead and OO stands for origin overhead) though it is on a microsecond scale.

Figure 5 shows the average computation overhead on the *Görli* blockchain network. The overhead fluctuates. It hits the summit of about 500 microseconds when the virtual machine executes about 10,000 transactions and comes down to 270 microseconds when 100,000 transactions come in. The performance difference is pretty apparent, and the statistics indicate that the highest performance gap between the two systems is around 40 microseconds when 10,000 transactions are executed.

Besides the computation overhead of the system, we also investigate the RPC communication latency added by VM-Studio. We use *gRPC* to transfer the RLP-encoded [22] transaction data between the Geth client and the virtual machine container. Figure 6 describes the average communication overhead of VM-Studio, which is caused by the data exchange between the *Geth* client and the virtual machine container. The communication overheads that VM-Studio adds when running on the Görli and Ropsten show downward trends. They start at about 2,000 microseconds when 1,000 transactions are executed and finally go down to about 400 microseconds. However, unlike the previous results, the overhead on the *BSC* stayed stable at about 300 microseconds during the whole

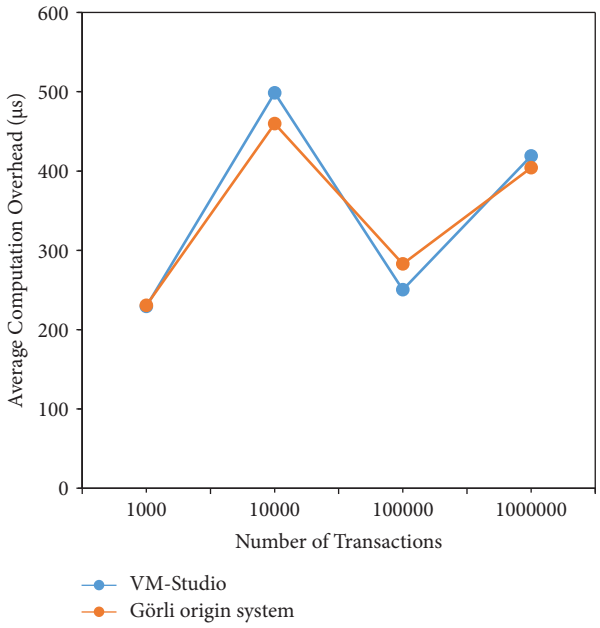


Figure 5: The average computation overhead on the *Görli* blockchain. The sampling size is fixed at 1,000,000 transactions, involving 1,806,736 blocks.

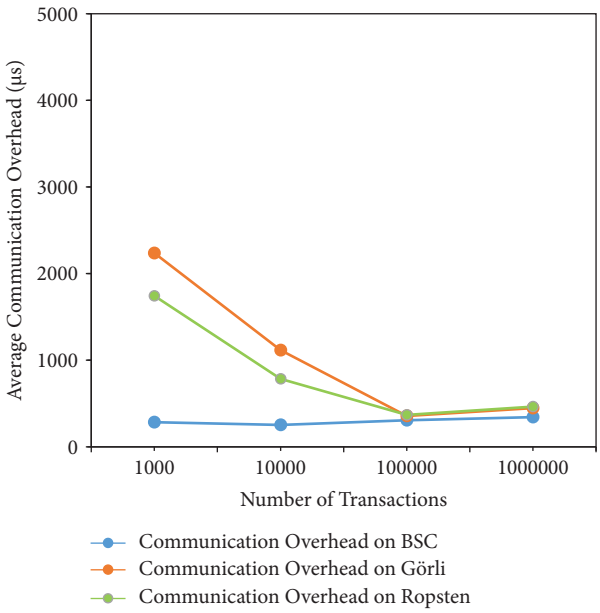


Figure 6: Extra communication overhead of VM-studio on three different blockchain networks.

procedure. The differences may mainly result from the characteristics of different blockchain systems, as the *Görli* and *Ropsten* are test networks that might contain some large transactions when they are first launched. While the *BSC* is the main network, its users will be more likely to consider the size of each transaction.

Finally, we put the computation overhead and the RPC communication overhead together to inspect the overall overhead of the VM-Studio. As we can see from Figure 7, the latency VM-Studio adds on continuously goes down with

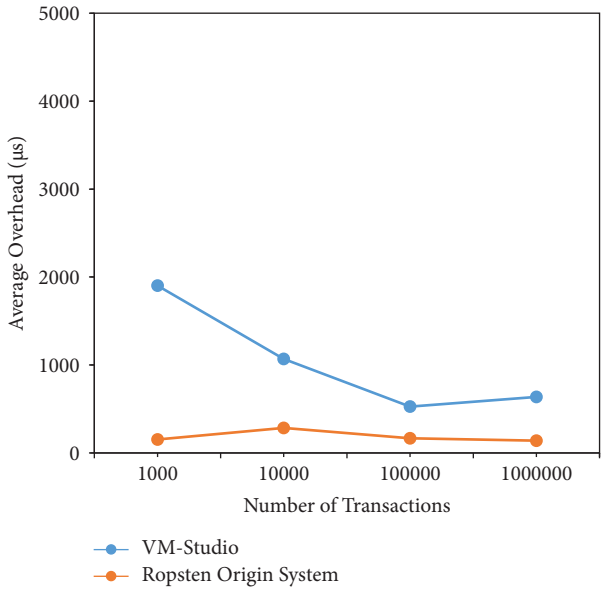


Figure 7: The average overhead on the *Ropsten* blockchain. The sampling size is fixed at 1,000,000 transactions, involving 402,361 blocks.

the increment of the transactions. The extra overhead stays about 450 microseconds when the 1,000,000 transactions are executed, mainly due to the RPC communication overhead that the VM-Studio adds. Furthermore, the situation on *Görli* is quite similar, which we can read in Figure 8, and the communication overhead is the main factor that affects the performance of the VM-Studio. As for the *BSC*, Figure 9 indicates that the performance of VM-Studio is quite close to that of the origin system. With about 300 extra microseconds, the difference between the performances of both systems is stable.

In conclusion, our experiments show that VM-Studio achieves availability. Due to the size of the transactions and the smart contract codes related to them, the prototype system's performance varies. As for communication costs, VM-Studio may add up to about 2000 microseconds to the origin system, mainly due to large transactions. Moreover, when economic factors restrict the sizes of transactions in the real-world production environment, the communication overhead is about 400 microseconds. Besides, the computation cost of the VM-Studio fluctuates around that of the original system. The maximal latency the VM-Studio adds is less than 40 microseconds, which is less significant compared with the communication overhead. Above all, the overhead added by the VM-Studio is majorly influenced by the communication overhead, which is about 400 microseconds in the real-world production environment. Therefore, we conclude that the extra latency is acceptable.

6. Discussion

6.1. Analysis of VM-Studio Scheme. The primary goal of VM-Studio is to ensure correct executions of both origin blockchain transactions and smart contracts. Here, as we

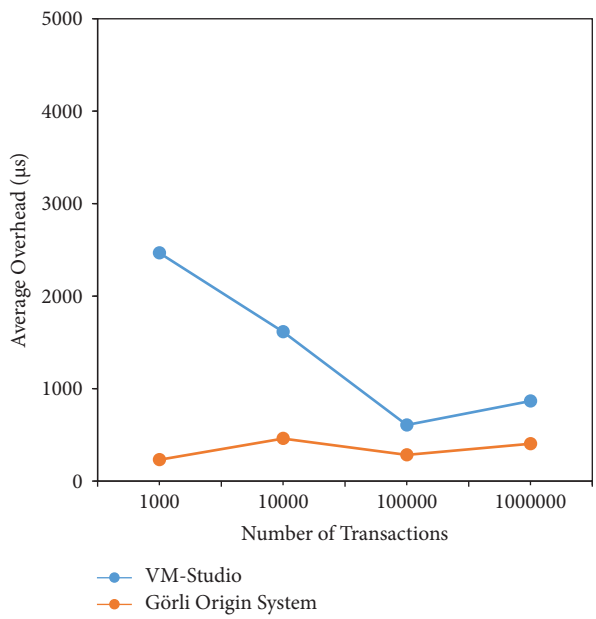


Figure 8: The average overhead on the *Görli* blockchain. The sampling size is fixed at 1,000,000 transactions, involving 1,806,736 blocks.

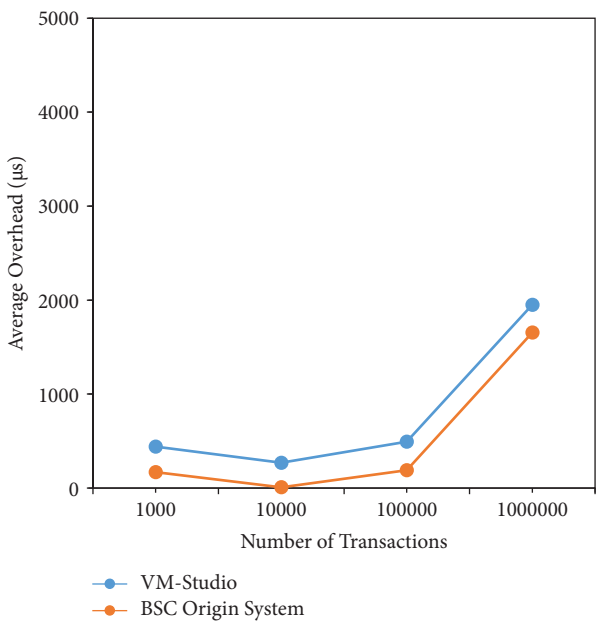


Figure 9: The average overhead on *BnB SmartChain*. The sampling size is fixed at 1,000,000 transactions, involving 1,311,838 blocks.

know, the execution of smart contracts is based on transactions. Therefore, we only discuss the correctness of transaction executions in the origin blockchain under our VM-Studio architecture.

We considered two factors in the architecture design of VM-Studio. First, the numbers and types of origin blockchains are various. VM-Studio is required to manage the resources of corresponding blockchains in a unified manner. We schedule the packaged containers into the container library for unified management to facilitate the overall migration or sharing of VM-Studio by nodes. Virtual machine re-encapsulation is not required; instead, the system offers unified interfaces to facilitate frequent fetching by the executor. In the database, containers provide the following unified interfaces: VM images, facilitating the first container loading and subsequent VM version checking and upgrading. Also, state snapshots are used by a VM to load data from the corresponding states and state database to import data into internal virtual machine images. Second, the origin blockchain transaction data are quite large, and each transaction will be sent to the container for execution as an atomic transaction. Therefore, the amount of transaction data multiplied by the number of origin blockchains will be millions. To solve the problem, we separate incoming messages of such atomic transactions from single-digit messages and place them in two different message queues. Thus, the cluster management software can allocate a large number of resources to the system for centralized processing of transactions without delaying the processing of other messages.

In addition, we have ensured this in the specific process design for four types of messages. First, when the message is input to VM-Studio for the first time, the system authenticates the packet header of the message through Algorithm 2. On the one hand, it verifies the signature of the message source. On the other hand, it dynamically adjusts the synchronization of the related origin blockchain in VM-Studio to facilitate subsequent loading processes. Second, VM images are loaded into containers from the origin blockchain. Transactions according to the original order have been continuously input into the container. The container's internal execution environment and that of the origin blockchain are entirely consistent, as long as the origin blockchain consensus has no objections to transaction results, which can ensure that transactions trading in VM-Studio perform correctly. Third, updating the virtual machine version will affect the transaction execution results. Since the update frequency of the virtual machine version is not high, we provide *MessageIV* to realize the update of the virtual machine in the database by VM-Studio.

6.2. Universality and Overhead. We explain that the VM-Studio solution is universal for the origin and target blockchains. Here, we illustrate two aspects. First, for an origin blockchain, the virtual machine means a machine that can automatically execute specific formats and certain types of transactions, and the relevant execution rules and verification rules have been hard coded inside the virtual machine. We load the virtual machine image into the container; thus, the container contains all the virtual machine rules, presenting corresponding data interfaces to the outside environment. Second, VM-Studio can be regarded as a set of components, which have little dependency on the blockchain architecture, and can be deployed at any node of the target blockchain. Therefore, VM-Studio also shows universality for the target blockchain. In addition, regarding the performance of VM-Studio to execute transactions on the target blockchain, in Section 6, experimental results have

shown that executing transactions on VM-Studio is slightly less efficient than those on the original system. The reason is that the transaction execution time related to sophisticated contract codes is mainly affected by the virtual machine, and the communication complexity inside and outside the container is insignificant in front of the computational complexity of the established transaction execution program of the virtual machine. However, the communication complexity mentioned above will take the lead when confronting simple transactions.

6.3. Read and Write Ability. We try to give a crosschain smart contract invocation example based on VM-Studio. Suppose that there exists a multichain system $M(n) = \{C_1, ..., C_{n-1}, C_n\}$, where $C(n)$ is a blockchain dedicated to initiating crosschain smart contract calls with VM-Studio components deployed on its nodes. While the VM images of $\{C_1, ..., C_{n-1}\}$ have been loaded into the VM-Studio container and their blockchain ledger, VM (latest version) images and state snapshots are stored in the VM-Studio database. At this point, we can consider that the blockchain C_n can run smart contracts on other chains.

As we know, calling a smart contract can be abstracted into two basic instructions: *Read and Write*. For a general blockchain system C, the usage of *Read* instruction only reads the chain state S_C but does not cause the change of S_C. Therefore, based on the world state of blockchain C and the execution environment of virtual machines, the *Read* instruction does not need to participate in the consensus of C to complete. However, the *Write* instruction directly changes the chain state S_C of the blockchain C. This process requires the consensus of chain C. Therefore, if a crosschain smart contract call transaction contains many *Write* instructions for different blockchain states, the system where VM-Studio is located is difficult to achieve. In particular, we specify that *Write* directives also include *Read* directives.

Now, consider a simple case where there is at most one crosschain smart contract call to a *Write* instruction. We give the following example:

(i) *Step 1.* Construct a crosschain smart contract call transaction, denoted as follows:

$$\left(id = id_{C_n}, \mathrm{Read}\left(C_1, C_2, \cdots, C_i\right), \mathrm{Write}\left(C(n)\right)\right). \quad (13)$$

The target chain of this transaction is $C(n)$, including chains $\{C_1, C_2, ...\}$, the *Read* instruction on C_I, and the *Write* instruction on the chain $C(n)$.

(ii) *Step 2.* Submit this transaction to VM-Studio, thus dividing it into $i + 1$ atomic transaction:

$$\mathrm{Read}\left(C_1\right), \cdots, \mathrm{Read}\left(C_i\right), \mathrm{Write}\left(C_n\right). \quad (14)$$

(iii) *Step 3.* The above $i + 1$ atom transactions are presented to cluster management sequentially, and the corresponding container is further invoked through the executor to execute previous i atom transactions.

(iv) *Step 4.* Call $\mathrm{Container}\left(VM\left(id_{C(n)}\right)\right)$ to perform $\mathrm{Write}\left(C_n\right)$.

(v) *Step 5.* Trade execution results and submit

$$\left(id = id_{C_n}, \mathrm{Read}\left(C_1, C_2, \cdots, C_i\right), \mathrm{Write}\left(C(n)\right)\right), \quad (15)$$

to the blockchain.

So far, we have achieved a simple single-write crosschain smart contract call transaction by VM-Studio in the heterogeneous chain environment. In the above transaction

$$\left(id = id_{C_i}, \mathrm{Read}\left(C_1, C_2, \cdots, C_i\right), \mathrm{Write}\left(C(n)\right)\right), \quad (16)$$

the transaction should be submitted to the blockchain C_i for confirmation after the consensus. However, this problem can be addressed if a VM-Studio component is used on blockchain C_i.

The invocation scheme of the crosschain smart contract with a multiwrite type needs to be realized by the locking mechanism and incentive mechanism under the premise of VM-Studio, starting from the atomicity of crosschain transactions. We will focus on this issue in the future.

7. Conclusion

The heterogeneity of blockchain is one of the significant factors hindering crosschain schemes. This study proposes VM-Studio, a universal crosschain smart contract verification and execution scheme. The main idea of VM-Studio design is to transform the compatibility and adaptation of the original transaction execution construction, namely, virtual machine construction, into the migration and encapsulation of origin blockchain virtual machines. By establishing a close virtual machine container and providing a unified data interface, the transaction execution environment of all VM-supported origin blockchains can be simulated on the target blockchain to complete the verification of crosschain smart contracts. Through theoretical analysis and experimental verification, we conclude that VM-Studio has negligible performance loss compared with the origin blockchain when executing transaction orders within the order of 100,000. Finally, we give an example of a single-write invocation towards crosschain smart contracts to demonstrate the feasibility and applicability of VM-Studio.

Data Availability

The data used to support the finding of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported by the National Key R&D Program of China through project 2020YFB1005600, the Natural Science Foundation of China through projects U21A20467,

61932011, and 61972019, the Beijing Natural Science Foundation through project M21031, and the Populus Euphratica Found CCF-Huawei BC2021009.

References

[1] S. Nakamoto, "Bitcoin: a peer-to-peer electronic cash system," *Decentralized business review*, vol. 21260, 2008.

[2] A. Kiayias, A. Russell, and B. David, "Ouroboros: a provably secure proof-of-stake blockchain protocol," in *Proceedings of the Annual International Cryptology Conference*, pp. 357–388, Santa Barbara, CA, USA, August 2017.

[3] R. Pass and E. Shi, "Thunderella: blockchains with optimistic instant confirmation," in *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pp. 3–33, Tel Aviv, Israel, April 2018.

[4] M. Campanelli, R. Gennaro, and S. Goldfeder, "Zero-knowledge contingent payments revisited: attacks and payments for services," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 229–243, New York, NY, USA, October 2017.

[5] Q. Wang, B. Qin, J. Hu, and F. Xiao, "Preserving transaction privacy in bitcoin," *Future Generation Computer Systems*, vol. 107, pp. 793–804, 2020.

[6] C. Schneidewind, I. Grishchenko, and M. Scherer, "Ethor: practical and provably sound static analysis of ethereum smart contracts," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pp. 621–640, Virtual Event, USA, November 2020.

[7] M. Wang and Q. Wu, "Lever: breaking the shackles of scalable on-chain validation," 2019, https://eprint.iacr.org/2019/1172.%202019.

[8] C. Li, P. Li, and D. Zhou, "A decentralized blockchain with high throughput and fast confirmation," in *Proceedings of the 2020 {USENIX} Annual Technical Conference (USENIX ATC)*, pp. 515–528, Boston, MA, USA, July 2020.

[9] Q. Wang and R. Li, "A weak consensus algorithm and its application to high-performance blockchain," in *Proceedings of the IEEE INFOCOM 2021-IEEE Conference on Computer Communications (INFOCOM)*, pp. 1–10, Vancouver, BC, Canada, May 2021.

[10] R. Belchior, A. Vasconcelos, S. Guerreiro, and M. Correia, "A survey on blockchain interoperability: past, present, and future trends," *ACM Computing Surveys*, vol. 54, no. 8, pp. 1–41, 2021.

[11] M. Herlihy, "Atomic cross-chain swaps," in *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing*, pp. 245–254, Toronto, ON, Canada, July 2018.

[12] B. T. C. Relay, *Bridge between the Bitcoin blockchain and Ethereum smart contacts*, 2018.

[13] F. Vogelsteller and V. Buterin, "Eip 20: erc-20 token standard," *Ethereum Improvement Proposals*, vol. 20, 2015.

[14] S. Noether and B. Goodell, "Triptych:logarithmic-sized linkable ring signatures with applications," *Data Privacy Management, Cryptocurrencies and Blockchain Technology*, pp. 337–354, Springer, Berlin, Germany, 2020.

[15] J. Xie, *Nervos CKB: A Common Knowledge Base for Crypto-Economy*, 2018.

[16] J. Poon and T. Dryja, *The Bitcoin Lightning Network: Scalable Off-Chain Instant payments*, 2016.

[17] J. Kwon and E. Buchman, *Cosmos whitepaper*, 2019.

[18] G. Wood, *Polkadot: Vision for a Heterogeneous Multi-Chain framework*, White paper, vol. 21, no. 2327, 2016.

[19] Z. Liu, Y. Xiang, and J. Shi, "Hyperservice: interoperability and programmability across heterogeneous blockchains," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, pp. 549–566, London, UK, November 2019.

[20] H. Wang, Y. Cen, and X. Li, "Blockchain router: a cross-chain communication protocol," in *Proceedings of the 6th International Conference on Informatics, Environment*, pp. 94–97, energy and applications, New York NY, USA, August 2017.

[21] N. Szabo, "Smart contracts: building blocks for digital markets," *Extropy: The Journal of Transhumanist Thought*, vol. 18, no. 2, 1996.

[22] G. Wood, "Ethereum: a secure decentralised generalised transaction ledger," *Ethereum project yellow paper*, vol. 151, pp. 1–32, 2014.

[23] E. Elrom, *Eos. Io Wallets and Smart Contracts*, The Blockchain Developer. Apress, Berkeley, CA, 2019.

[24] E. Androulaki, A. Barger, and V. Bortnikov, "Hyperledger fabric: a distributed operating system for permissioned blockchains," in *Proceedings of the Thirteenth EuroSys Conference*, pp. 1–15, New York, NY, USA, July 2018.

[25] S. Thomas and E. Schwartz, *A Protocol for Interledger payments*, 2015.

[26] L. Gudgeon, P. Moreno-Sanchez, and S. Roos, "Sok: layer-two blockchain protocols," in *Proceedings of the International Conference on Financial Cryptography and Data Security (FC)*, pp. 201–226, New York, NY, USA, May 2020.

[27] A. Garoffolo, D. Kaidalov, and R. Oliynykov, "Zendoo: a zk-SNARK verifiable cross-chain transfer protocol enabling decoupled and decentralized sidechains," in *Proceedings of the 2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, pp. 1257–1262, Singapore, December 2020.

[28] A. Back, M. Corallo, and L. Dashjr, "Enabling blockchain innovations with pegged sidechains," vol. 72, pp. 201–224, 2014, http://www.opensciencereview.com/papers/123/enablingblockchaininnovations-with-pegged-sidechains.

[29] H. Abbas, M. Caprolu, and R. Di Pietro, "Analysis of polkadot: architecture, internals, and contradictions," in *Proceedings of the 2022 IEEE International Conference on Blockchain (Blockchain)*, pp. 61–70, Espoo, Finland, August 2022.

Hindawi
Security and Communication Networks
Volume 2022, Article ID 7144516, 13 pages
https://doi.org/10.1155/2022/7144516

Research Article

Security Analysis on Blockchain-Powered Mobile APPs Connected with In-Vehicle Networks by Context-Based Reverse Engineering

Xingyu Wu,[1] Ziyan Qiao,[2] Xingjuan Cai,[1] Qian Wang,[1] Zhiqiang Xie,[2] Rui Sun,[2] Dong Zi,[2] Wenjia Niu,[2] and Endong Tong[2]

[1]*School of Computer Science and Technology, Taiyuan University of Science and Technology, Taiyuan 030024, China*
[2]*Beijing Key Laboratory of Security and Privacy in Intelligent Transportation, Beijing Jiaotong University, Beijing 100044, China*

Correspondence should be addressed to Endong Tong; edtong@bjtu.edu.cn

Received 24 June 2022; Accepted 29 August 2022; Published 23 September 2022

Academic Editor: Yujue Wang

The controller area network (CAN) bus for interconnection of electronic control units (ECUs) plays a highly important role in modern intelligent vehicles. To facilitate the CAN Bus accessing to vehicle control or diagnosis, a number of mobile APPs are designed and published by automobile manufacturers to support driving and vehicle-based social network, and some are realized through the in-vehicle infotainment (IVI) middleware. Blockchain technologies are also mature for automobiles to interact service information with the whole industry. Unfortunately, there is a serious threat of command leakage from these mobile APPs, and the reverse engineering (RE) can be exploited by hackers. Previous work has researched this threat by an automatic reverse engineering tool on both automotive android and IOS APPs. However, in such common tool, APP itself-related contexts, including the feature information of CAN Bus commands, vehicle application functions, and control diagnostic protocols, are overlooked, which might be utilized to promote the reverse engineering recall. In this paper, we propose a context-based reverse engineering approach to find deep hidden commands for further revealing security threats for blockchain-powered mobile automotive APPs. For the reverse engineering, we design a context model of four-order tensor to organize multidimensional contexts and establish a continuous updating mechanism. Based on the context model, we further develop two basic analysis algorithms, max-compute (A) and clustering (A), to perform the analysis of CAN Bus commands. Extensive experiments are conducted, and we evaluate it by two metrics, recovered ratio and correctness ratio. Experimental results and the case studied on the familiar APP Carly validate the effectiveness of our approach and reveal the threat of command leakage.

1. Introduction

Nowadays, with the development of information technology (e.g., IoT and AI), we have entered an era of intelligent vehicles. More automotive manufacturers focus on introducing more sensors with edge computing of computers such as electronic control units (ECUs) and advanced intelligent control systems into vehicles [1–3]. More specifically, those vehicle control functions ranging from steering, braking, acceleration, to lighting and infotainment are controlled by a variety of ECUs, which requires the support of an efficient infrastructure to ensure the momentary connection of in-vehicle networks. Actually, controller area network (CAN) [2] is the most widely deployed network which provides a shared internal network. For its implementation, CAN Bus is developed as a multi-master broadcast communication protocol and offers advantages such as self-diagnosing, error correction [2], and autonomous driving by utilizing predefined commands for specific automobiles. The recent study suggests that the typical luxury sedan generally integrates 50–70 ECUs with thousands of commands. Therefore, for future intelligent

automobiles, the CAN Bus and its commands essentially play a particularly important role to represent the technology advances of a modern vehicle and its manufacture.

Generally, there are two types of connection to the CAN Bus: onboard diagnostics (OBD-II) [4] connection and middleware connection. The OBD connection is a direct physical access to the CAN Bus via OBD port (typically under the dash), which can be plugged with a dongle. Among different types of dongles on the market, some only provide diagnosis or monitoring functions, while others offer remote control functionality [4] in addition to diagnosis capabilities. The middleware connection is implemented to connect the CAN Bus by introducing middleware technologies which resides in the vehicle's head unit, and it is the major equipment to drive the system of in-vehicle infotainment (IVI) [5]. However, both of the connections to CAN Bus are transparent for drivers or common users and they only need to operate via APP installed in mobile phone or IVI system. Specifically, dongles can be accessed by mobile phone APP via Bluetooth, Wi-Fi, or cellular network, and head unit can be accessed by IVI APP via direct network communication. Moreover, mobile phone APPs situate in IOS or Android environment, and IVI APPs situate in middleware with software module to extend mobile applications features to vehicles, typically such as MirrorLink [5], Android Auto [6], and CarPlay [7]. For instance, the Siri of CarPlay can be adapted to make calls and interact with other applications of CarPlay.

Mobile phone APPs and IVI APPs, middleware of head unit, and CAN Bus form a wide attack surface on the automobile by spoofing or injecting CAN Bus commands. As a result, hackers can stop the engine remotely and disable the brake. Many efforts summarized the vulnerabilities of remote control for vehicles through this attack surface, including mobile APP masquerading, privilege escalation, replay attack [8], DoS or DDoS attack, and man-in-the-middle attack. Some emerging work focuses on reverse engineering (RE) [9] of the CAN Bus commands, which is regarded as a significant building block for subsequent attacks on in-vehicle systems. In addition to observing the traffic inside the automotive for obtaining the CAN packets and replaying them back into the CAN to attack the vehicle, Wen et al. [4] developed a cost-effective (no real car needed) and automatic (no human intervention required) system CAN-HUNTER for reverse engineering of CAN Bus commands, which use just car companion mobile APPs for both Android and iOS platforms. Inspired by their efforts, we re-examine their implementation as shown in Figure 1, and we take the APP Carly used in their experiment as the instance. Through the reverse engineering, we get the source code including two CAN Bus command IDs, a label of Unified Diagnostic Services (UDS) protocol, a parameter with value 1A87, as well as a text description "left-front door" with semantics to show a corresponding function. However, it is difficult to further reveal the detailed command in the real CAN Bus only based on such source code. It requires massive contexts on other aspects, characteristics of CAN Bus message, third-party or automotive APP function, control/diagnose protocol, and vehicle details, except source code itself.

In addition, the implementation of blockchain technology [10, 11] on automobile industry is promising to manage information and interaction. Many researches try to exploit blockchain technology [12] to share information such as manufacturer data, real-time data from sensors, and environment variables. It can be also correspondingly used on automotive mobile APPs to support more innovations. Therefore, it is important to do the research blockchain-powered mobile APPs.

In this paper, we propose an approach of context-based reverse engineering, aiming to establish a general framework to perform security analysis on the phone and IVI APPs toward the threat of CAN Bus command leakage.

In summary, we make the following contributions:

(i) We design a context model of four-order tensor with a continuous updating mechanism through the interaction between the source code and the RE analysis module to organize multidimensional contexts for reverse engineer

(ii) We further develop two basic analysis algorithms—max-compute (A) and clustering (A), to explore the semantics of CAN Bus commands

(iii) We evaluate our approach by two metrics, recovered ratio (Recover@1, Recover@3, Recover@5) and correctness ratio on the three different dimensions of the model and receive impressive results

The remainder of this paper is organized as follows. Section 2 describes the background. Then, we propose the context-based reverse engineering of IVI APPs in Section 3. Experiments and detailed analysis are reported in Section 4. Section 5 discusses the related work. Finally, we conclude the paper in Section 6.

2. Background

2.1. Communication Infrastructure in Modern Automobile. Figure 2 presents the communication framework toward the automobile CAN Bus, and there are three ways of connection: (1) using mobile APP for accessing OBD-II dongle to connect CAN Bus; (2) using mobile APP to directly connect to the IVI APP for further CAN Bus access; (3) using mobile APP to connect the cloud server via cellular network and then further access the IVI APPs from the cloud for indirectly connecting to the CAN Bus. We can discover that the wireless communication protocol includes cellular network such as 2G~5G, Wi-Fi, and Bluetooth, while the communication between IVI APP and CAN Bus implemented directly by middleware. The middleware implements a middle interface between mobile applications and vehicle ECUs. More specifically, the middleware enables the drivers' interaction with IVI and even displays the compatible mobile applications completely on the IVI touch screen. In general, modern middleware is implemented in head unit and supported by most automobiles. Once the request is received by the middleware, the vehicle will take

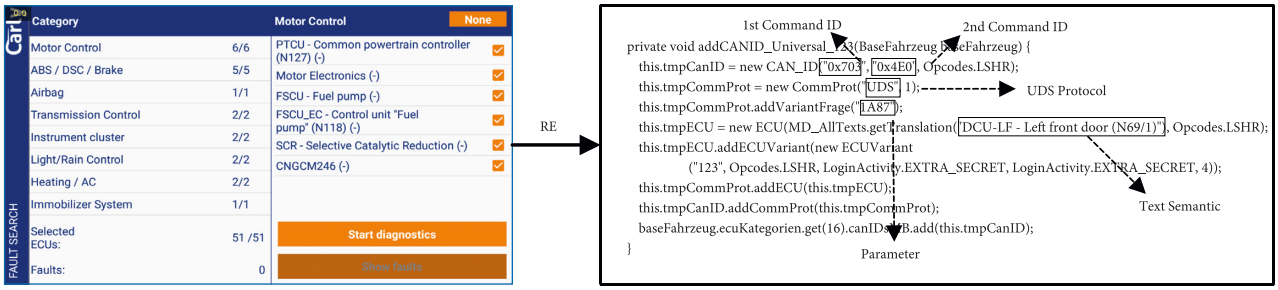


Figure 1: An example of mobile APP reverse engineering.

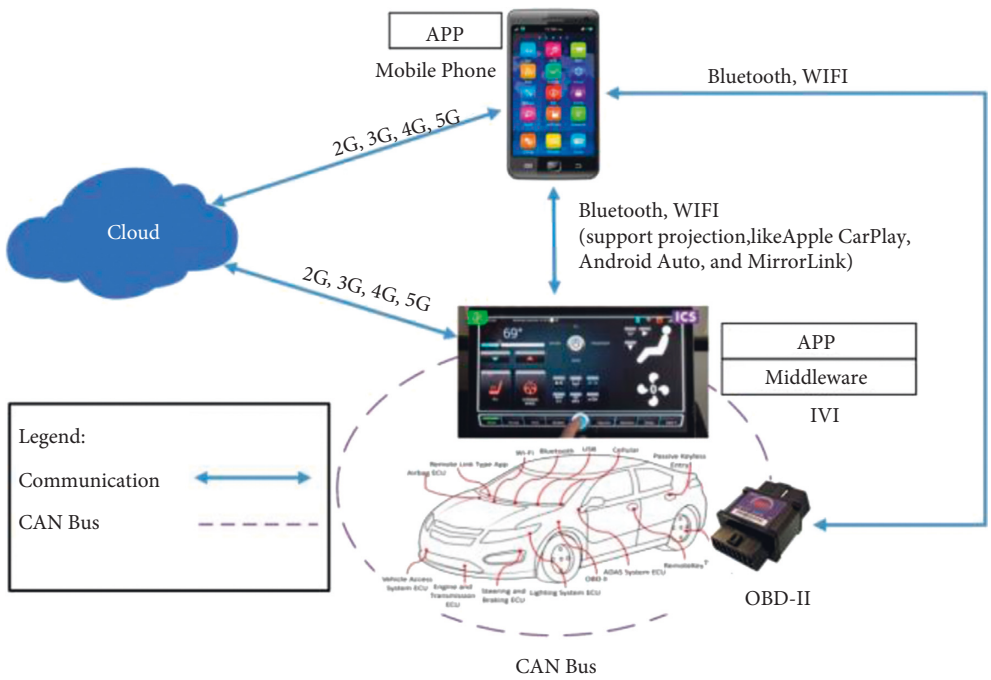


Figure 2: The automobile CAN Bus-oriented communication framework.

corresponding actions and send back a response or notification across the CAN Bus.

Note that the middleware utilizes the technologies of remote projection and procedural calls (RPC). Typically, APPs such as Apple CarPlay, Android Auto, and MirrorLink use projection to display an adapted user interface on the vehicle IVI screen.

2.2. CAN Bus Command. The CAN Bus, typically consisting of transmitting and receiving amplifier, is designed to connect different ECUs. The data which are sent back and forth for CAN communications are defined as message or frame. As shown in Figure 3, a CAN Bus message is composed with a specific data structure.

The onset of the frames is indicated by a SOF (start of frame) package (a dominant bit) with EOF (end of frame) package (7 recessive bits) at the frame end. CAN frame carries data containing at most 8 bytes, along with segments for characterizing the message identifier, the CRC (cyclic redundancy check) check, the RTR (remote transmission request), the IDE (identifier extension bit), the r0, and the DLC (data length code). The message identifier has either 11 or 28 bits and refers to the target ECU to forward. The CRC field consists of 16 bits, where one bit is delimiter and others are for checksum. ACK indicates that whether the data are received normally.

In this paper, differently, we separate the term "command" from "CAN message identifier (CAN ID)," because all ECUs will receive the command in network Bus, but only the specified ECU can execute the function as a response. Under the context of OBD and UDS protocol, some service ID (SID in UDS) or PID in OBD composing data in the frame can represent an explicit command for requesting a function of ECUs. Hence, the CAN Bus command exists within the 0–8 bytes in data segment of a CAN Bus message, and it is referred as a CAN Bus command or a command segment. The command syntactic which is hexadecimal is generally classified into two categories, (1) control command, e.g., unlocking the right-rear door or stopping the

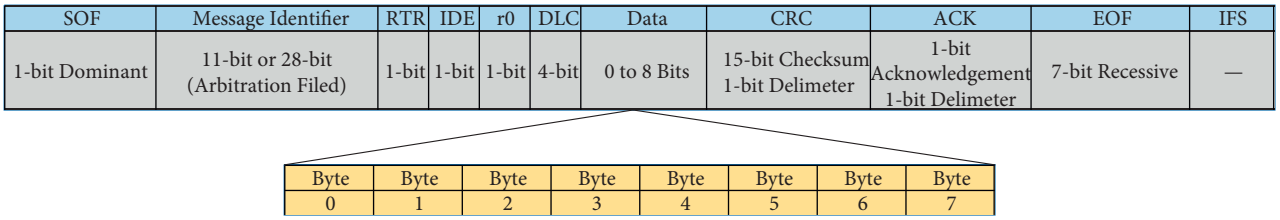


Figure 3: Structure of CAN frame for bus command understanding.

Table 1: Data captured in the CAN Bus of a real sedan.

SeqNum	System time	CAN channel	CAN ID	LEN	Command
0	21:00.9	ch1	$0 \times 00AF$	0×08	00 00 00 00 00 00 00 00
1	21:00.9	ch1	0×0169	0×06	7B B1 40 FE 24 00
2	21:00.9	ch1	$0 \times 01F2$	0×02	20 00

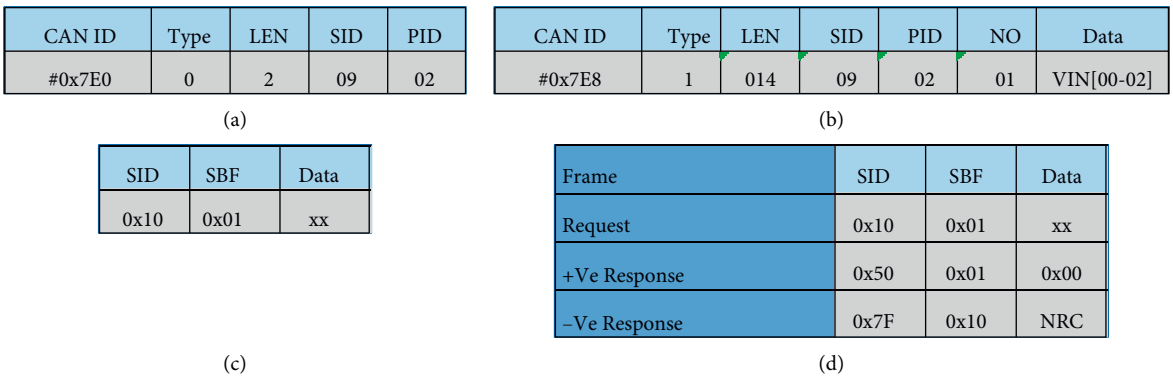


Figure 4: Request and response frames of OBD and UDS. (a) OBD-II request frame. (b) OBD-II response frame. (c) UDS request frame. (d) UDS response frame (positive and negative response).

engine and (2) diagnosis command for querying necessary status data.

Table 1 shows three records about the key commands of CAN message captured from the CAN Bus of a real sedan. The LEN field indicates the total number of bytes in the response. The three commands have 2, 6, 8 bytes, respectively, and take specific functions (e.g., open the left-front door) for specific ECUs according to corresponding CAN IDs.

2.3. OBD and UDS Frame Structure. The request and response frames of OBD and UDS are shown in Figure 4. In Figure 4(a), the LEN field specifies the following byte number in the request. SID is the service identifier, which is 0×09 in this case and represents the "Request Vehicle Data" service with a parameter ID (PID). The PID 0×02 corresponds to the vehicle identification number (VIN). The response for the request frame is shown in Figure 4(b). The SID and PID code fields should be the same as the request frame. The type is 0 if the response satisfies the request within a single frame, while the type 1 is used to indicate the "start frame" of a multi-frame packet if the response includes multiple frames. In an OBD response, the SID field equals 0×40 plus the SID from the request. NO is the number of data items (1 for the VIN in this case) for service 0×09. Data segment contains the first three bytes of the requested data.

In addition to legitimate OBD, many vehicles also support the newer Unified Diagnostic Services (UDS) standard, defined by ISO 14229–3, which builds on legislated OBD. The UDS protocol working on the CAN protocol can request the maximum 8 byte data and get the response from a message. In UDS protocol, there are also two available types of frames—diagnostic request frame and diagnostic response frame. The UDS protocol frame format is shown in Figures 4(c) and 4(d). The request frame has 3 fields including SID, sub-function ID (SFD), and data. There are two types of response, positive response and negative response. Whenever the tester requests to the server, it will send the response message by adding 0×40 to the respective SID for reference, if it is correct and the server has executed the request successfully. In positive response, the first byte should be request SID plus 0×40. If the client requests in an inappropriate frame format or the server is not able to execute the request due to the internal problem, then it will send a negative response to the client. The first byte of negative response should be $0 \times 7F$, and the second byte and the third byte should be SID and response code, respectively. If the ECU or server fails to send a request, it will send a response message with negative response code (NRC).

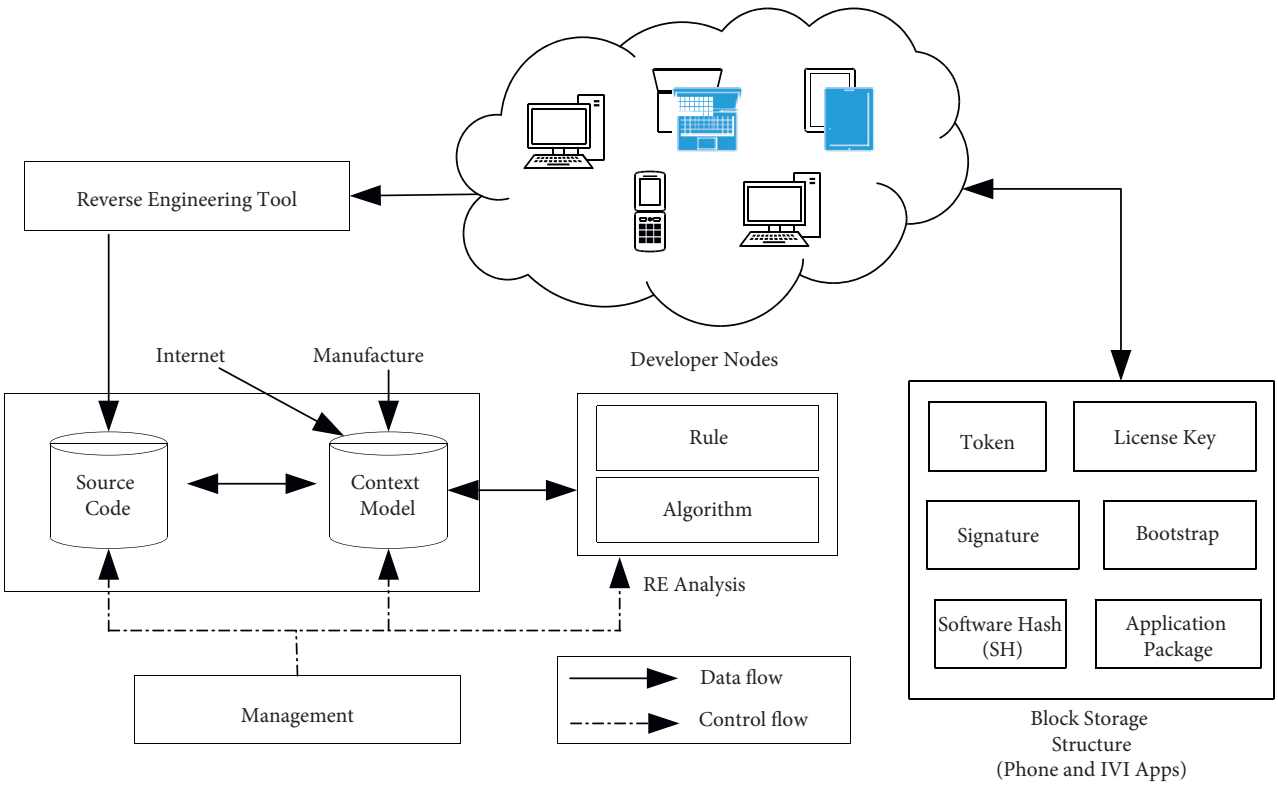


Figure 5: The workflow of context-based reverse engineering on blockchain-powered mobile APP.

3. Design

3.1. Model Architecture. The whole workflow of context-based reverse engineering on blockchain-powered vehicle APP is shown in Figure 5. This method is designed for reverse engineering on the phone and IVI APPs, suitable for IOS and Android platforms.

In our approach, the traditional APPs are combined with blockchain which has multiple software developer nodes [13]. We used the alliance blockchain, which is managed by all the alliance members, and all nodes in this alliance chain can share on-chain data. Those nodes are uploaded and downloaded with the block storage structure including some symbols and data of phone and IVI APPs. The acquirement of source data directly correlates with blockchain nodes. Firstly, we use typical RE tools including Apktool [13], dex2jar [14], and jadx [15] to obtain source codes. Next, we build a context model to organize multidimensional knowledge to support RE analysis. Note that, the context model has a continuous updating mechanism of the interaction with the source code and the RE analysis module, which contains two parts, rules and algorithms for analyzing. The management module is responsible for coordinating the source code, context model, and RE analysis and forming a complete loop to work continuously.

3.2. Constructing 4-Order Tensor-Based Context Model. Based on APP permissions, CAN IDs, semantic, and command, we design a 4-order tensor model to obtain and store semantics of command function described by text (see

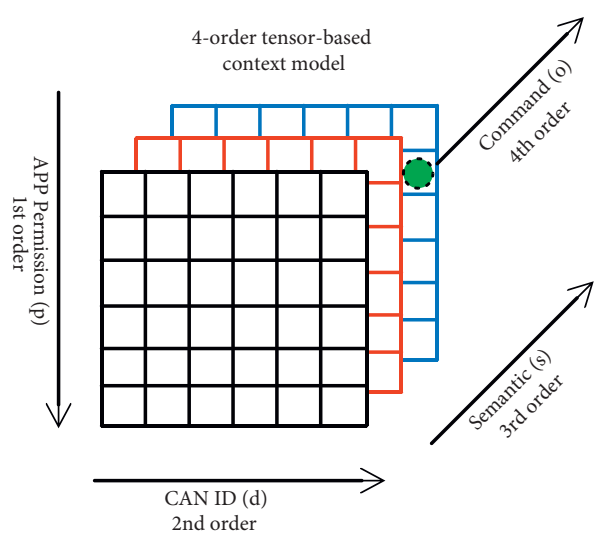


Figure 6: 4-order tensor-based context model for analyzing CAN Bus command.

Figure 6). We construct a 4-order tensor $A \in R^{X \times Y \times Z \times M}$ ($X = |p|$, $Y = |d|$, $Z = |s|$, $M = |o|$) in which $A_{x,y,z,m}$ refers to an element with the coordinate of (x, y, z, m) in the tensor A.

For the tensor A, the 1st-order p describes the APP permissions, such as access_wifi_STATE and write_external_STORAGE; The 2nd-order d contains the CAN ID; the 3rd-order s distinguishes the different semantics of commands, for example, "RWTS—rear-end door closing" refers to the semantic corresponding to the command

"0 × 4F7"; the 4th-order o stores 8-byte commands. Under the index of 4 orders, we can locate a text description for the semantic of command function. To facilitate the computation based on the tensor A, we further obtain the following four matrices in

$$A(p) =, A_{i,\mathbf{y},\mathbf{z},\mathbf{m}} = \left(A_{i,1,1,1}, \ldots, A_{i,1,1,|\mathbf{o}|}, \ldots, A_{i,1,|s||o|,}, \ldots, A_{i,|d|,|s|,|o|}\right), 1 \le \mathbf{i} \le |p|, \mathbf{i} \in N^{+}. \tag{1}$$

Here, the column vector size is $|p| \times 1$ and within the first-order p, we can choose i to determine the APP permission as an index to corresponding vector $A_{i,y,z,m}$.

We further get matrices $A(d)$, $A(s)$, and $A(o)$ with the same way in the following formulae:

$$A(\mathbf{d}) =, A_{\mathbf{x},i,\mathbf{z},\mathbf{m}} = \left(A_{1,i,1,1}, \ldots, A_{1,i,1,|\mathbf{o}|}, \ldots, A_{1,i,|\mathbf{s}|,|\mathbf{o}|}, \ldots, A_{|p|,i,|\mathbf{s}|,|\mathbf{o}|}\right), 1 \le \mathbf{i} \le |d|, \mathbf{i} \in N^{+}. \tag{2}$$

Here, the column vector size is $|d| \times 1$ and within the 2nd-order d, we can choose i to determine the CAN ID as an index to corresponding vector $A_{x,i,z,m}$.

$$A(\mathbf{s}) =, A_{\mathbf{x},\mathbf{y},i,\mathbf{m}} = \left(A_{1,1,i,1}, \ldots, A_{1,1,i,|o|}, \ldots, A_{1,|d|\,i,|o|}, \ldots, A_{|p|,|d|,i,|o|}\right), 1 \le \mathbf{i} \le, |s|\mathbf{i} \in N^{+}. \tag{3}$$

Here, the column vector size is $|s| \times 1$ and within the 3rd-order s, we can choose i to determine the semantics as an index to corresponding vector $A_{x,y,i,m}$.

$$A(\mathbf{o}) =, A_{\mathbf{x},\mathbf{y},i,\mathbf{m}} = \left(A_{1,1,1,i}, \ldots, A_{1,1,|s|,i}, \ldots, A_{1,|d|,|s|,i}, \ldots, A_{|p|,|d|,|\mathbf{s}|,i}\right), 1 \le \mathbf{i} \le |o|, \mathbf{i} \in N^{+}. \tag{4}$$

Here, the column vector size is $|o| \times 1$ and within the 4th-order o, we can choose i to determine the commands as an index to corresponding vector $A_{x,y,z,i}$.

3.3. Reverse Engineering Analysis. In this section, we propose two basic analysis algorithms; one is named max-compute (A), which makes a statistic on the number of revealed semantics in three orders. Another is named clustering (A), which performs clustering on texts corresponding to CAN Bus command, so as to analyze similar functions.

In Algorithm 1 line 2, for each element in APP permission p, it computes the number of revealed semantics *Amount*. Finally, we output the vector of $List\_p(A(p)[|p|]$, $List\_d(A(d)[|d|]$, $List\_o(A(o)[|o|]$.

For the given semantic set s, a text clustering method corresponding to CAN Bus command is realized to get the commands with the most similar function. As Algorithm 2 shows, this method begins with $|c|$ clusters. In line 5, we calculate distance between different clusters. In line 7, we merge two individual clusters with the shortest distance into a larger cluster. We then repeat the operations from line 4 to line 8 until the number of cluster reaches $|k|$.

The distance between two individual clusters is calculated by $Dal(c_i,c_j)$.

$$\%\ \mathrm{Dal}(c_i, c_j) = \frac{1}{|C_{\mathbf{i}}||C_{\mathbf{j}}|} \sum_{s_i \in C_{\mathbf{i}}} \sum_{s_i \in C_{\mathbf{j}}} L(s_i, s_j), L(s_i, s_j) = \frac{\left|\{w_k | w_k \in s_i \cap w_k \in s_j\}\right|}{\log(|s_i|) + \log(|s_j|)}, \tag{5}$$

where c_i and c_j are clusters and $c_i \cap c_j = \varnothing$, $L(s_i, s_j)$ is the similarity between two semantics $s_i \in c_i$ and $s_j \in c_j$. w_k represents a word in a semantic, $s = w_1, w_2, \ldots, w_n$. Molecular part calculates the numbers of the same words that appear simultaneously in both semantics, and the denominator part calculates the logarithmic sum of the number of words in the semantics.

3.4. Computational Complexity Analysis. Computational complexity analysis of the two analysis algorithms is discussed as follows. We firstly analyze the computational complexity of Algorithm 1, which traverses all states in the three-dimensional space $p \times d \times s$ to calculate the number of revealed semantics in three orders. Thus, the computational complexity of Algorithm 1 is a level of $O(n)$, where n is the size of semantics.

For each cluster x, $x \in \{1, 2, \ldots, |k|\}$, as we need to perform clustering until the current number of clusters greater than the terminated cluster number $|k|$ and calculate the distance between any two commands. The computational complexity for computing the distance is $O(mn)$, in which the parameter m refers to the size of o_i and the parameter n refers to the size of o_j. Therefore, the computational complexity of Algorithm 2 is a level of $O(|k||k|mn)$.

4. Experiment

4.1. Setup. The experimental environment configuration is shown in Table 2, based on the Windows 11 of a Honor Hunter V700 laptop with Intel(R) Core(TM) i7-10750H CPU @ 2.60 GHz, 16G RAM. There are three reverse engineering tools used to analyze in our experiment. Apktool [13] is used to extract permission information from AndroidManifest.xml and original byte codes of the APPs. It is a tool for reverse engineering 3rd party, closed, and binary Android APPs which can decode resources to original form nearly and rebuild them after making some modifications. We use dex2jar [14] to work with files android.dex and java.class and convert the file classes.dex to classesdex2jar.jar which is the combination of original source class files. Jadx [15], a standalone graphical utility, is used to display Java source codes from Java object code ".class" files.

4.2. Evaluation Metric

4.2.1. Recovered Ratio. We describe the semantic integrity discovered by our method in terms of recovered ratio, which counts the number of semantics. We define the corresponding recovered ratio R_p, R_d, R_o for the three orders p, d, and o as follows, where $|\text{List}\_p|$, $|\text{List}\_d|$, and $|\text{List}\_o|$ represent the number of semantics using Algorithm 1 on the orders p, d, and o, respectively. The number of semantic recoveries with App permission as the tensor is represented by $|S_p|$, the number of semantic recoveries with CAN ID as the tensor is represented by $|S_d|$, and the number of semantic recoveries with command as the tensor is represented by $|S_o|$. R represents the semantic recovered ratio, $|S\_\text{recovered}|$ is the number of recovered semantics, and $|S\_\text{real}|$ is the real number of semantics. We further define $\mathscr{R}@3$ and $\mathscr{R}@5$ which take the best top 3 and 5 experimental results supposing that the results are arranged in order indexed by i.

$$\mathscr{R}_{\mathbf{p}} = \frac{\text{Maxcompute}\left(A_{i,\mathbf{y},\mathbf{z},\mathbf{m}}\right).|\text{List}\_p|}{|S_p|},$$
$$\mathscr{R}_{\mathbf{d}} = \frac{\text{Maxcompute}\left(A_{\mathbf{x},i,\mathbf{z},\mathbf{m}}\right).|\text{List}\_d|}{|S_d|}, \tag{6}$$
$$\mathscr{R}_{\mathbf{o}} = \frac{\text{Maxcompute}\left(A_{\mathbf{x},\mathbf{y},i,\mathbf{m}}\right).|\text{List}\_o|}{|S_o|},$$

$$\mathscr{R}_{\mathbf{p}}@3 = \frac{\sum_{i=1}^{3}|\text{List}\_p|}{3|S_p|},$$
$$\mathscr{R}_{\mathbf{d}}@3 = \frac{\sum_{i=1}^{3}|\text{List}\_d|}{3|S_d|}, \tag{7}$$
$$\mathscr{R}_{\mathbf{o}}@3 = \frac{\sum_{i=1}^{3}|\text{List}\_o|}{3|S_o|},$$

$$\mathscr{R}_{\mathbf{p}}@5 = \frac{\sum_{i=1}^{5}|\text{List}\_p|}{3|S_p|},$$
$$\mathscr{R}_{\mathbf{d}}@5 = \frac{\sum_{i=1}^{5}|\text{List}\_d|}{3|S_d|}, \tag{8}$$
$$\mathscr{R}_{\mathbf{o}}@5 = \frac{\sum_{i=1}^{5}|\text{List}\_o|}{3|S_o|},$$

$$\mathscr{R} = \frac{|S_{\text{recovered}}|}{|S_{\text{real}}|}. \tag{9}$$

4.2.2. Correctness Ratio. To measure the effectiveness of our method, we define the correctness ratio $\mathscr{C}$ that calculates the number of correct semantics in the third-order s, where $c(A_{x,y,i,m}) = 1$ when $A_{x,y,i,m}$ is a true semantic; otherwise, $c(A_{\mathbf{x},\mathbf{y},i,\mathbf{m}}) = 0$. $\mathscr{C}_p$, $\mathscr{C}_d$, $\mathscr{C}_o$, and $\overline{\mathscr{C}}$, respectively, shows the correctness ratio on the order p, d, o and the average correctness ratio.

$$\mathscr{C}_p = \frac{\sum_i c_p\left(A_{x,y,i,m}\right)}{|S_{\text{recovered}}|}, \mathscr{C}_d = \frac{\sum_i c_d\left(A_{x,y,i,m}\right)}{|S_{\text{recovered}}|}, \mathscr{C}_o = \frac{\sum_i c_o\left(A_{x,y,i,m}\right)}{|S_{\text{recovered}}|}, \tag{10}$$

$$\overline{\mathscr{C}} = \frac{\sum_i c_{\mathbf{p}}\left(A_{\mathbf{x},\mathbf{y},i,\mathbf{m}}\right) + \sum_j c_{\mathbf{d}}\left(A_{\mathbf{x},\mathbf{y},j,\mathbf{m}}\right) + \sum_k c_{\mathbf{o}}\left(A_{\mathbf{x},\mathbf{y},k,\mathbf{m}}\right)}{3|S_{\text{recovered}}|}. \tag{11}$$

4.3. Case Study on Carly. Our entire workflow on Carly APP is shown in Figure 7, and detailed processes are described as follows.

4.3.1. Decompilation. We firstly disassemble the Android application code and convert it into intermediate languages, such as Jimple format or Smali format. Different types of code transformation are implemented according to the specific analysis tools. Based on these decompiled codes, we perform a further analysis.

4.3.2. Construction of Application Code Graph. Based on static analysis technology, we analyze the calling relationship among functions in Android application code and build the function call graph (FCG) in which each point in FCG represents a function and each edge represents the calling relationship among functions. The calling relationships are described once the FCG is established. Further, for each function in the Android application code to construct control flow graph (CFG), each point in CFG represents a continuous code block and each edge represents a possible branch execution path relationship, such as those branches caused by control instructions if and switch.

4.3.3. Execution Path Search. In both FCG and CFG, the search algorithm is utilized to find all possible execution paths targeting the critical function calling behavior. Firstly, according to selected key functions, it selects all the calling sequences of those functions by searching the paths on the FCG; then, search all the control flow conditions related to the key function according to the CFG of the function in each calling sequence and obtain the corresponding paths which also needs to make sure their acyclicity.

4.3.4. Execution Control. Modify the control flow conditions in each selected execution path to ensure the correct execution process of application following the selected path. Modify the entry function of the Android software, and turn it directly to the selected execution path in above step. Then, modify all the execution flow conditions on the path to ensure that the key function will eventually be called. This process can be done with code insertion technology, and each path will produce a new application.

Input: 4-order tensor A
Output: ($List\_p(A)$ [1], ..., $List\_p(A)[|p|]$), ($List\_d(A)$ [1], . . ., $List\_d(A)[|d|]$), ($List\_o(A)$ [1], . . ., $List\_o(A)[|o|]$)

```
for each p ∈ p:
    //the number of revealed semantics in p order
    Amount = SemanticNum(A(p));
        List_p(A)[p].add(Amount);
end for
return List_p(A)[|p|]
for each d ∈ d:
    //the number of revealed semantics in d order
    Amount = SemanticNum(A(d));
        List_d(A)[d].add(Amount);
end for
return List_d(A)[|d|]
for each o ∈ o:
    //the number of revealed semantics in o order
    Amount = SemanticNum(A(o));
        List_o(A)[o].add(Amount);
end for
return List_o(A)[|o|]
```

Algorithm 1: Max-compute (A).

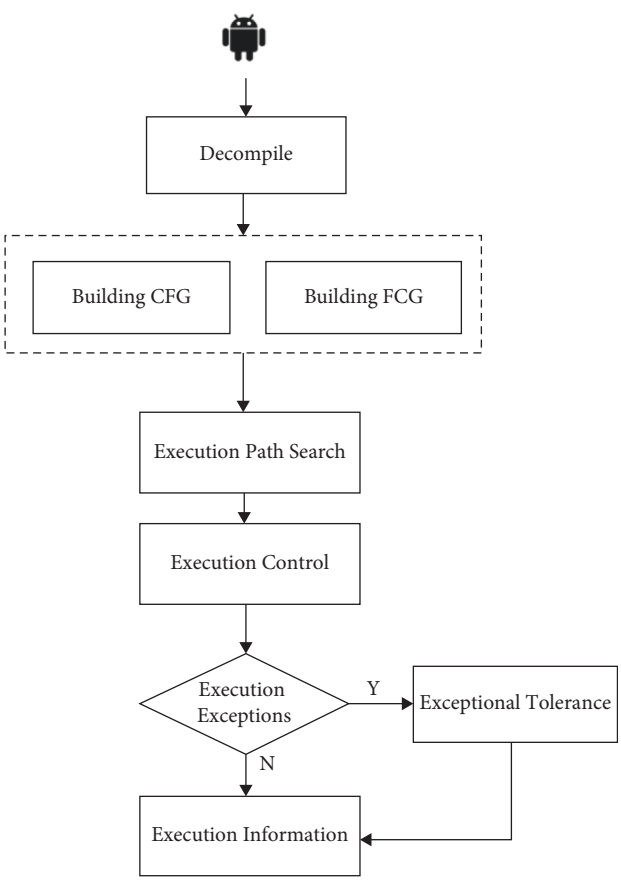


Figure 7: The workflow of reverse analysis on Carly APP.

4.3.5. Dynamic Execution of Exceptional Tolerant. There are many execution exceptions during running when the execution process of application is controlled and some execution conditions are modified. The application software will stop executing immediately if those exceptions are not processed. To guarantee that the code for selected path can be executed, the exception-handling logic of the Android execution environment is modified so that execution proceeds from the next instruction even though the error also continues. For Java code, modify the Android source code to change the exception-handling process in the Dalvik virtual machine execution environment and tolerate the exception to execute from the next instruction. For C/C++ code, similar methods can be used to tolerate exceptions or errors based on virtualization techniques.

4.3.6. Execution Parameter Collection. During the application execution process, the required parameters are collected for further analysis. According to different collected execution parameters, the data are output when the application software is executed, such as the ID identifier data for a CAN message.

The numbers of APP permission, CAN ID, and command and part of contents in Carly are shown in Table 3 in which there are huge numbers of CAN ID and command. In Table 4, we list correspondence between semantic and vehicle models. The first column indicates some semantics, and the second line shows 13 models of Mercedes-Benz. Different models may have different sets of semantics. For example, all of car models have the semantic AB—airbag, model C Stufenheck/Kombi-203 has the semantic WSS—weight sensing system, while *G* Steilheck/Cabrio—463 does not.

4.3.7. Evaluation on Recovered and Correctness Ratio. The results of recovered ratio are shown in Table 5, and we respectively calculate the value from the three dimensions by equations (7)–(10). There is the highest recovery rate on the dimension of APP permission, followed by CAN ID and

Input: semantic $s = \{s_1, s_2, \ldots, s_{|s|}\}$, cluster number $k = \{k_1, k_2, \ldots, k_{|k|}\}$
Output: $\{c_1, c_2, \ldots, c_{|k|}\}$
(1) **repeat**
(2) $x \leftarrow 1$
(3) //initial setting
$c_x = \{c_{x;1}, \ldots, c_{x;|o|}\} = \{\{s_1\}, \ldots, \{s_{|s|}\}\}$
(4) **repeat**
(5) //initial cluster merging
$(c_\alpha, c_\beta) = \text{argmin}_{c_i \in c_x, c_j \in c_y} \text{Dal}(c_i, c_j)$
$= \text{argmin}_{c_i \in c_x, c_j \in c_y}, (1/|C_i||C_j|) \sum_{s_{i \in C_i}} \sum_{s_{i \in C_j}} L(s_i, s_j)$
(6) $New\_c = c_\alpha \cup c_\beta$
(7) $c_x = (c_x \backslash \{c_\alpha, c_\beta\}) \cup \{New\_c\}$
(8) **until** $|c_x| \geq k_x$
(9) $x \leftarrow x + 1$
(10) **until** $x > |k|$
(11) **return** $\{c_1, c_2, \ldots, c_{|k|}\}$

Algorithm 2: Clustering (A).

Table 2: Experimental environment configuration.

Experimental environment	Environmental configuration
Operating system	Windows 11
CPU	Inter(R) Core(TM) i7-10750H CPU @ 2.60 GHz
RAM	16G
Software	Apktool 2.6.1, Dex2jar 2.1, Jadx 1.4.1

command which has a worse result comparably. On the dimension of APP permission, the recovered number of semantics is determined by the other two dimensions CAN ID and command from our context model. Firstly, because both of them have large cardinal numbers 2460×20010, small absence of semantics has little effect on the overall recovery rate. In addition, as shown in CAN frame of Figure 3, CAN ID and command are almost able to determine one semantic except for some special semantics. Therefore, the Recover@1 is up to 99.8%. The huge cardinal numbers also lead to the less decay of Recover@3 and Recover@5 in APP permission. Similarly, when we calculate the rate from the dimension of CAN ID or command, the value is determined by APP permission and command or APP permission and CAN ID. However, on the one hand, one permission may be mapped to multiple semantics that makes it difficult to judge which is the correct semantic the permission mapped. On the other hand, only CAN ID or command cannot uniquely ascertain one semantic. That leads to the worse results in CAN ID and command. The size of APP permission × command is 14×20010, while APP permission × CAN ID is 14×2460. Therefore, the result in command has greater impact which is 66.7% because the smaller cardinal number and its decline trend from Recover@1 to Recover@5 are also the greatest. Overall, we conclude that we can recover the most semantics from the dimension of APP permission or even CAN ID and we obtain a good result of the average recovered ratio which is 84.24%. We also evaluate the correctness ratio by equations (7) and (8). The result validates the effectiveness of our context model that all of the ratios are close to 100%.

Table 3: Part of dimensional tensors in context model.

Tensor type	Content	Number
APP permission	WRITE_EXTERNAL_STORAGE BLUETOOTH_ADMIN BLUETOOTH INTERNET ACCESS_NETWORK_STATE WAKE_LOCK ACCESS_COARSE_LOCATION READ_EXTERNAL_STORAGE ACCESS_WIFI_STATE ...	14
CAN ID	$0 \times 7E0$, $0 \times 7E8$ $0 \times 6A3$, $0 \times 4D4$ 0×632, 0×486 $0 \times 63B$, $0 \times 5BB$ $0 \times 62A$, 0×485 ...	2460
Command	7B B1 40 FE 24 00 00 00 80 53 20 00 00 00 00 00 0E 00 00 00 08 63 00 00 8A 8 F 10 06 FD 68 8B 01 8B 10 00 03 FF 00 00 48 53 31 38 37 37 30 36 12 05 07 10 11 6E 00 00 ...	20010

4.4. Analysis of Cluster Semantics. The clustering result of semantic is shown in Table 6, where we set the threshold $|k| = 176$. We, respectively, calculate the normalized maximal distance, minimal distance, and mean distance in the

Table 4: Part of distribution among semantics on different vehicle models.

Semantic	Vehicle model												
	A class—176	B class—245	B class—246	C Stufenheck/ Kombi—203	CLS coupe—219	GLK Steilheck—X204	G Steilheck/ Cabrio—463	S limousine—221	AMG/GT roadster—199	SL roadster—231	M Steilheck—164	E coupe/ Cabrio—207	CLA coupe—117
AB-airbag	√	√	√	√	√	√	√	√	√	√	√	√	√
CGW—central gateway	√	√	√	√	√	√	√	√	√	√	√	√	√
TV—TV tuner	√		√		√	√		√					√
WSS—weight sensing system				√	√	√		√	√		√	√	
HS—seat heater					√						√		
SGR—radar sensor control unit						√		√	√	√		√	
FLAL—fire extinguishing system (N100)							√						
TPM—tire pressure monitor (N88)				√	√	√		√	√				
SBC—sensotronic brake control									√				
FSW—windshield wiper (M6/1)	√	√	√	√	√	√	√	√	√	√	√	√	√
KG—keyless go	√		√					√	√	√	√		√
360 camera—60 camera				√	√	√			√			√	
KAB—control unit “camera cover” (N42)													√

three dimensions, in which distance is referred to the average value of $Dal(c_i, c_j)$ between pairwise semantics across all the semantics using equations (5) and (6). For the complete data of semantic in the last line, the max distance represents the distance between the two farthest data in our context model which is up to 1, while the min distance is 0.014 and the mean distance is 0.603. The maximal mean distance 0.893 appears in the dimension of APP permission that indicates it is widely distributed with its data in this dimension. On the dimension of command, its mean distance is smaller that means it has a concentrated distribution. Therefore, this clustering result shows that the data of semantic are decentral or diverse on the dimension of APP permission that concludes most of the data while those are concentrative on the dimension of command that only concludes part of data.

5. Related Work

5.1. Attack Vectors of CAN Security Analysis. As an essential part of modern automobile, CAN is responsible for coordinating the various and sophisticated ECUs of the vehicle to control sub-systems like steering, braking, doors, and windows properly. It is also exploited to develop many applications, such as remote control, vehicle diagnosis, security monitoring, vehicle hacking, and autonomous driving, which has improved driving safety, comfort, and functionality. However, it also causes new attack surfaces to the modern automobile. In recent years, there has been an increasing amount of studies on the CAN Bus attacks. The mobile phone APPs and IVI APPs, middleware of head unit, and CAN Bus form a lot of attack vectors to the automobile by spoofing or injecting CAN Bus commands. There are six categories, mobile APP masquerading, privilege escalation, replay attack, compromising attack, DoS or DDoS attack, and man–in-the-middle attack.

Mobile APP masquerading: The attackers could clone or repackage the legitimate mobile applications straightforwardly by reverse engineering of Android APPs or iPhone APPs [16]. They release the masqueraded APPs with malicious logic on Google Play or Apple App Store, which is not easy to detect [17–21] and obtain vehicle status, manipulate vehicle functionalities, and impact vehicle safety. Privilege escalation: Some middleware APIs which need to be opened to developers are quite security-sensitive. Therefore, it is necessary to audit such critical APIs before invocation. Otherwise, malicious application could invoke these APIs covertly. A malicious application X can successfully invoke the critical APIs by leveraging a flawed design of an open application Y, which has the privilege to use security-sensitive APIs. The study of Han et al. [18] found that Apple private Object C function calls could be compromised by using dynamic loading of functions during runtime. Replay attack: Attackers intercept the communication in one session and retransmit the messages in another session to launch the relay attack. For example, the permissions can be intercepted and reused in malicious application by attackers. The middleware API requests from the legitimate application can be intercepted and resent to the middleware for receiving services. Compromising attack: Attackers exploit the vulnerabilities on both Android system [22] and IOS [18] system to manipulate the mobile applications running on the mobile systems in various ways, such as intercepting the middleware API, replaying, and revising the communication between the application and middleware. Meanwhile, the head unit system could be compromised due to it runs in a more privileged mode and it is vulnerable for remote exploitation. It has been demonstrated that it is feasible to hack into head unit system directly or remotely [2, 9]. DoS or DDoS attack: The attackers could launch the DoS against mobile application by leveraging the compromised mobile system or head unit system to discard the APIs or reject response for the APIs. Also, the middleware APIs could be invoked by the attackers who can query vehicle status and flood messages for vehicle CAN Bus to interfere the vehicle control. In addition, the attackers can launch DDoS attack against vehicle manufacturer facility by controlling a large amount of zombie machines [23] and then make the middleware functionality fail. Man-in-the-middle attack: There are multiple parties in the modern automobile, including vehicle manufacturer facility, mobile application, and vehicle head unit. Attackers could launch the attack in the interaction between any two parties to intercept the encrypted credential of users or obtain the permissions of invoking APIs and reuse such permissions in their own applications. Considering all of the studies reviewed here, it is undoubted that APP reverse engineering is one of the most important techniques to launch a remote attack.

5.2. APP Reverse Engineering. Several previous studies [2, 3, 24–26] have shown that reverse engineering of CAN Bus commands can remotely attack a vehicle. Through a number of possible attack surfaces [5, 9, 27] like Bluetooth, Internet, and APPs, attackers could obtain the CAN packets and replay them back into the CAN to attack the vehicle. Koscher et al. [2] demonstrated that an attacker could observe and reverse-engineer CAN packets, and later inject new packets to induce various attacks. In another major study, Miller and Valasek [24] reported some of attacks via the CAN Bus. An attacker could acquire codes running on ECUs via an attack over Bluetooth, telematics, tire sensor, and physical access to re-send packets and thus to affect the regular work of the automobile. The study by Miller and Valasek [27] identified techniques for reverse engineering CAN Bus commands and demonstrated that attackers could manipulate CAN-enabled components of an automobile. Miller and Valasek [26] in their study showed a remote attack that could form physical control of some aspects of the vehicle by reverse engineering the mechanics tools, ECU firmware, etc. Recently, Li et al. [3] proposed a cost-effective (no real car needed) and automatic (no human intervention required) system, CAN-HUNTER, to realize the reverse engineering of CAN Bus commands. This system only utilizes the companion mobile APPs without using realistic automobiles, and it could perform well in both Android and IOS platforms.

Table 5: Evaluation results of recovered ratio on three dimensions.

Tensor	Recovered ratio		
	Recover@1 (%)	Recover@3 (%)	Recover@5 (%)
APP permission	99.8	99.6	99.5
CAN ID	87.6	87.1	86.9
Command	66.7	65.8	65.2
Total		84.24	

Table 6: Clustering result of semantic using Algorithm 2 ($|k| = 176$).

Dimension	Max distance	Min distance	Mean distance
APP permission	0.992	0.019	0.893
CAN ID	0.986	0.021	0.775
Command	0.994	0.023	0.648
Overall	1	0.014	0.603

However, there are some problems in above researches. For instance, Li et al. [3] had a lack of considering the context. For overcoming it, this work designs a context model of four-order tensor to organize multidimensional contexts trying to avoid the security threat from APP itself-related context.

6. Conclusions

In order to uncover the security threat that overlooking vehicle mobile APP itself-related context, this paper proposes a context-based reverse engineering approach to locate deep hidden commands. We construct a context model with 4-order tensor, including APP permission, CAN ID, semantic, and command, to organize multidimensional contexts, and we also build a continuous updating mechanism. Two algorithms max-compute (A) and clustering (A) are proposed to support RE analysis. We perform the experiment based on the Carly APP with several RE tools and evaluate our approach by two indexes, recovered ratio (Recover@1, Recover@3, Recover@5) and correctness ratio to assess the degree of recovered semantics in different dimensions. The high average recovered ratio and correctness ratio show the effectiveness of our method in the two evaluation metrics. Therefore, there are certain security risks that can be utilized to reverse engineering recall and vehicle mobile APP should pay attention to itself-related context such as the feature information of CAN Bus commands, vehicle application functions, and control diagnostic protocols.

This work is expected to conduct a series of studies on the safety of blockchain automotive mobile APP, including but not limited to (1) research on instruction-semantic mining techniques, (2) matching CAN Bus instruction and corresponding context semantics, and (3) constructing reinforcement learning model toward reverse process.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grant nos. 61972025, 61802389, 61672092, U1811264, and 61966009 and the National Key R&D Program of China under Grant nos. 2020YFB1005604 and 2020YFB2103802.

References

[1] P. Mundhenk, *Security for Automotive Electrical/electronic (E/E) architectures [M]*, Cuvillier Verlag, Göttingen Germany, 2017.

[2] K. Koscher, A. Czeskis, F. Roesner, P. Shwetak, and K. Tadayoshi, "Experimental Security Analysis of a Modern Automobile," in *Proceedings of the 2010IEEE symposium on security and privacy*, pp. 447-462, Oakland CA USA, May 2010.

[3] L. Li, J. Liu, L. Cheng et al., "CreditCoin: a privacy-preserving blockchain-based incentive announcement network for communications of smart vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 7, pp. 2204–2220, 2018.

[4] H. Wen, Q. Zhao, and Q. A. Chen, "Automated Cross-Platform Reverse Engineering of CAN Bus Commands from mobile Apps," in *Proceedings of the 2020 Network and Distributed System Security Symposium (NDSS'20)*, San Diego California USA, February 2020.

[5] S. Mazloom, M. Rezaeirad, and A. Hunter, "A Security Analysis of an {In-Vehicle} Infotainment and App Platform," in *Proceedings of the 10th USENIX Workshop on Offensive Technologies (WOOT 16)*, Austin TX, June 2016.

[6] A. K. Mandal, F. Panarotto, A. Cortesi, P. Ferrara, and F. Spoto, "Static analysis of Android Auto infotainment and on board diagnostics II apps," *Software: Practice and Experience*, vol. 49, no. 7, pp. spe.2698–1161, 2019.

[7] D. L. Strayer, J. M. Cooper, M. M. McCarty et al., "Visual and cognitive demands of carplay, android auto, and five native infotainment systems," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 61, no. 8, pp. 1371–1386.

[8] M. Bozdal, M. Samie, and I. Jennions, "A survey on can Bus protocol: attacks, challenges, and potential solutions," in *Proceedings of the 2018 International Conference on Computing, Electronics & Communications Engineering (ICCECE)*, pp. 201-205, Southend UK, August 2018.

[9] S. Checkoway, D. McCoy, B. Kantor, and B. Anderson, "Comprehensive experimental analyses of automotive attack surfaces," in *Proceedings of the 20th USENIX Security Symposium (USENIX Security 11)*, USA, August 2011.

[10] R. M. Parizi and H. Sajad, "A Blockchain-Based Framework for Detecting Malicious mobile Applications in App Stores," in *Proceedings of the IEEE Canadian Conference of Electrical*

and Computer Engineering (ICCECE), pp. 1–4, Edmonton AB Canada, May 2019.

[11] W. Wang, J. Song, G. Xu, Y. Li, H. Wang, and C. Su, "ContractWard: automated vulnerability detection models for ethereum smart contracts," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1133–1144, 2021.

[12] M. M. Moharrer and S. D. Nogoorani, "A Decentralized App Store Using the Blockchain Technology," in *Proceedings of the 17th International ISC Conference On Information Security And Cryptology (ISCISC)*, pp. 14-21, Tehran Iran, September 2020.

[13] "Apktool", "A tool for reverse engineering Android apk files," https://ibotpeaches.github.io/Apktool.

[14] "Dex2jar", "Tools to work with android .dex and java," https://sourceforge.net/projects/dex2jar.

[15] "Jadx", "Dex to Java decompiler," https://github.com/skylot/jadx.

[16] W. Enck, "A study of android application security," *USENIX security symposium*, vol. 2, no. 2, 2011.

[17] N. Viennot, E. Garcia, and J. Nieh, "A Measurement Study of Google Play," in *Proceedings of the the 2014 ACM International Conference on Measurement And Modeling of Computer Systems*, pp. 221-233, New York NY USA, 2014.

[18] L. Han, A. L. Kashyap, T. Finin, and J. Mayfield, "UMBC_EBIQUITY-CORE: semantic textual similarity systems," in *Proceedings of the Second Joint Conference on Lexical and Computational Semantics (∗ SEM)*, vol. 1, pp. 44-52, August 2013.

[19] X. Liu, J. Liu, S. Zhu, W. Wang, and X. Zhang, "Privacy risk analysis and mitigation of analytics libraries in the android ecosystem," *IEEE Transactions on Mobile Computing*, vol. 19, no. 5, pp. 1184–1199, 2020.

[20] W. Wang, Y. Li, X. Wang, J. Liu, and X Zhang, "Detecting Android malicious apps and categorizing benign apps with ensemble of classifiers," *Future Generation Computer Systems*, vol. 78, pp. 987–994, 2018.

[21] W. Wang, X. Wang, D. Feng, J. Liu, Z. Han, and X. Zhang, "Exploring permission-induced risk in android applications for malicious application detection," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 11, pp. 1869–1882, 2014.

[22] Y. T. Lee, W. Enck, and H. Chen, "{PolyScope}:{Multi-Policy} Access Control Analysis to Compute Authorized Attack Operations in Android Systems," in *Proceedings of the 30th USENIX Security Symposium (USENIX Security 21)*, pp. 2579-2596, Auguest 2021.

[23] W. Wang, Y. Shang, Y. He, Y. Li, and J. Liu, "BotMark: automated botnet detection with hybrid analysis of flow-based and graph-based traffic behaviors," *Information Sciences*, vol. 511, pp. 284–296, 2020.

[24] C. Miller and C. Valasek, "Adventures in automotive networks and control units," *DefCon*, vol. 21, no. 260-264, pp. 15–31, 2013.

[25] J. Staggs, *How to Hack Your Mini cooper: Reverse Engineering Can Messages on Passenger Automobiles*, Institute for Information Security, Japan, 2013.

[26] C. Miller and C. Valasek, *Remote Exploitation of an Unaltered Passenger Vehicle*, Black Hat USA, 2015.

[27] C. Miller and C. Valasek, *A Survey of Remote Automotive Attack Surfaces*, Black Hat USA, 2014.

Hindawi
Security and Communication Networks
Volume 2022, Article ID 1607996, 17 pages
https://doi.org/10.1155/2022/1607996

Research Article

NSSIA: A New Self-Sovereign Identity Scheme with Accountability

Qiuyun Lyu,[1] **Shaopeng Cheng**,[1] **Hao Li**,[1] **Junliang Liu**,[2] **Yanzhao Shen**,[1] **and Zhen Wang**[1]

[1]*School of Cyberspace, Hangzhou Dianzi University, Hangzhou, Zhejiang 310018, China*
[2]*Security Department, Hangzhou Meichuang Technology Co, Ltd., Hangzhou, Zhejiang 310011, China*

Correspondence should be addressed to Yanzhao Shen; yanzhaoshen@hdu.edu.cn

Received 29 April 2022; Revised 16 July 2022; Accepted 26 July 2022; Published 5 September 2022

Academic Editor: Dawei Zhao

Self-sovereign identity (SSI) is a new distributed method for identity management, commonly used to address the problem that users are lack of control over their identities. However, the excessive pursuit of self-sovereignty in the most existing SSI schemes hinders sanctions against attackers. To deal with the malicious behavior, a few SSI schemes introduce accountability mechanisms, but they sacrifice users' privacy. In addition, the digital identities (static strings or updatable chains) in the existing SSI schemes are as inputs to a third-party executable program (mobile app, smart contract, etc.) to achieve identity reading, storing and proving, and users' self-sovereignty are weakened. To solve the above problems, we present a new self-sovereign identity scheme to strike a balance between privacy and accountability and get rid of the dependence on the third-party program. In our scheme, one and only individual-specific executable code is generated as a digital avatar-i for each human to interact with others in cyberspace without a third-party program, in which the embedding of biometrics enhances uniqueness and user control over their identity. In addition, a joint accountability mechanism, which is based on the shamir (t, n) threshold algorithm and a consortium blockchain, is designed to restrict the power of each regulatory authority and protect users' privacy. Finally, we analyze the security, SSI properties and conduct detailed experiments in terms of the cost of computation, storage, and blockchain gas. The analysis results indicate that our scheme resists the known attacks and fulfills all the six SSI properties. Compared with the state-of-the-art schemes, the extensive experiment results show that the cost is larger in server storage, blockchain storage, and blockchain gas, but is still low enough for practical situations.

1. Introduction

Identity management (IdM) has experienced increased interest due to the ever-growing demand for digital identities, as people become overly dependent on online services [1]. However, each traditional IdM system usually adopts centralized authorization, authentication and maintains identity data independently [2]. As a result, enormous online IdM services force people to manage a large number of digital identities, which leads to the problem of identity fragmentation [3] and is vulnerable to identity attacks, such as identity impersonation, privacy leakage, and identity fraud [4]. Even worse, users are lack of control and ownership over their digital identities in the traditional IdM [5, 6]. Therefore, a distributed method for identity management called self-sovereign identity (SSI) is proposed [7], in which the users are central to the administration of identities. Fortunately, the rise of distributed ledger technology (DLT), such as blockchain, has also made it possible to construct self-sovereign identities [8–10]. In comparison to the centralized management used by the traditional IdM, SSI schemes shift decision authority to users through secured DLT [11] and allow them to possess full control over their identities and data [12–16].

According to the goals to achieve, existing SSI schemes can be divided into the following three categories: junior SSI schemes [17–21], SSI schemes with sybil-resistance [22–26], and SSI schemes with accountability [27, 28]. To give users' control over their identities and data, junior SSI schemes adopt DID standards [17], smart contracts [18, 19], or credential chain [21]. Static strings, such as DIDs, addresses of smart contracts, or updatable chains are employed to

identify the users. However, the fact that users can hold as many identities as they want facilitates the implementation of sybil attacks. Therefore, many scholars introduced additional certificate authority [22] and biometrics [23–26] to ensure that each user has one and only DID-based digital identity in their *SSI* schemes with sybil-resistance. But unfortunately, the above schemes cannot reveal the identities of malicious users. To deal with the problem, SSI schemes with accountability [27, 28] are proposed. Since users are represented by credential chains in [27], a regulatory authority checks the malicious credentials with the personal information in a central registry to identify the malicious users. However, the audit of malicious users initiated by a single regulatory authority may lead to serious problems of inadequate regulation or injustice. Different from the scheme [27], the problems caused by a single regulation are overcome in the paper [28]. Specifically, the sanctions lists and a fuzzy matching method based on secure multiparty computation are applied to identify the credentials of suspicious users. However, both the central registry [27] and the sanctions lists [28] inevitably leak user privacy.

On the other hand, metaverse, as the evolving paradigm of the next generation of the Internet [29], will contain enormous amounts of applications and bring new challenges to the SSI. Metaverse is considered as a massive virtual environment parallel to the physical world, in which users interact through digital avatars [30]. That is, digital avatars are executable programs that own and control their identities for the user's physical self [29, 30]. However, the digital identities (static strings or updatable chains) in the existing SSI schemes are all used as inputs to a third-party executable program (mobile app, smart contract, etc.) to achieve identity reading, storing, and proving. Thus, the existing SSI schemes cannot play well in metaverse and also weaken users' self-sovereignty. In detail, the dependence on a third-party executable program during the usage of SSI inevitably leads to the problems of a single point of failure and privacy leakage.

Inspired by the digital avatars in metaverse, and taking the above problems in the existing SSI schemes into account, we propose a new self-sovereign identity scheme with accountability. And the contributions of the proposed scheme are summarized as follows:

(i) We propose a new self-sovereign identity scheme with accountability (NSSIA), in which executable code is introduced to allow users to control their identities completely and the balance between privacy and accountability is achieved.

(ii) To get rid of the dependence on third-party programs, one and only individual-specific executable code is distributed to each user, where the user's biometrics are embedded to enhance uniqueness and user control. The hash of the executable code is used as an identifier and each user can use his/her own local executable code to store, read, and prove identities with network servers. For simplicity, the term "digital avatar" in metaverse is borrowed and reformed to "digital avatar-i" to denote the executable code focusing on digital identity.

(iii) In order to regulate malicious users fairly without violating privacy, a joint accountability mechanism is introduced to decentralize the power of regulatory authorities and hide users' information in reality through shamir(t, n) threshold signature algorithm, while the impartial audit is further guaranteed by a consortium blockchain.

(iv) We analyze the proposed scheme in detail in terms of security, SSI properties in the generation phase and conduct extensive experiments in the cost of computation, storage, and blockchain gas.

The rest of this article is organized as follows. Section 2 introduces the related work of SSI schemes. In Section 3, the system model, security model, and design goals are introduced and Section 4 describes our scheme in detail. We analyze our proposed scheme in terms of security and performance in Sections 5 and 6, respectively. Finally, the conclusion and future work are given in Section 7. This paper has been published as an arxiv preprint [31].

2. Related Work

2.1. Junior SSI Schemes. Junior SSI schemes [17–21] are first proposed to allow users to control their own identities. To enable users to have full control over their identities, Takemiya and Vanieiev [17] designed a security protocol for storing encrypted personal information based on Hyperledger Iroha. The decentralized identifier (DID) [32] was used as the unique identifier of each user, while entries that characterized a user's identity were represented in the form of verifiable claims [33]. For self-sovereignty, all the claims were stored locally on user's phone in encrypted form. Different from Takemiya and Vanieiev [17], smart contracts were used to represent the user's identity in the paper of [18, 19]. Concretely, they both designed a kind of smart contracts with addresses as identifiers, specifically for managing identities. Once published, these contracts were owned by the corresponding users. And, the user's identity information was stored in IPFS [18] and stored in the user's device in the form of a Merkle tree [19] for self-sovereignty. However, both DIDs and smart contract addresses are machine-readable static strings, which are difficult for users to understand, leading to the dilemma of managing digital identities.

Then, a decentralized service architecture for self-sovereign social communication, proposed by Westerkamp et al. [20], solves the above problem. In this scheme, the user's identifier was represented as a human-readable name which was generated by the smart contract-based Ethereum Name Service (ENS). Besides, the user's data was stored in his/her own API server, and the Uniform Resource Identifier (URI) of the server was stored and linked to the human-readable name on the blockchain. However, such human-readable identifier is still inherently static, which is easily impersonated by malicious users during use.

Fortunately, this problem can be alleviated by a general provable claim model proposed in the paper [21]. With reference to the structure of the blockchain, the self-sovereign identity was designed as a growing chain of user's claims. And, the user's identity could be used only after the authentication of the verifier on the existing claims, thus alleviating the risk of identity being impersonated. But, due to the lack of necessary authentication before identity registration, users can create as many identities as they want, which facilitates the implementation of sybil attacks.

2.2. SSI Schemes with Sybil-Resistance. In order to let each user has one and only digital identity, SSI schemes with sybil-resistance [22–26] are proposed. A commitment scheme combined with zk-SNARK was introduced in [22] to provide integrity and privacy of user information simultaneously. In this scheme, to ensure integrity and avoid reuse, only after the user's information and the corresponding commitment were confirmed by the CA, a certificate would be issued to the user. And during usage, user's data was encrypted by zk-SNARK to prevent privacy leakage. However, the verification of the commitment by the CA can only guarantee the integrity of the information from the user, not the authenticity, which means the CA can be deceived by false information.

For the authenticity and reliability of user identity, biometric identification is introduced into SSI schemes [23–26]. In 2018, Othman and Callahan [25] designed a novel method for decentralized biometric-based self-sovereign identity. In this scheme, the user's identity was created based on the DID specification. Also, in order to associate each user with their own identity, biometrics (fingerprint, face, voice, etc.) were encrypted and stored in the corresponding DID document. But unfortunately, biometric information is only collected through the user's mobile app, which presents an opportunity for adversaries to commit identity fraud.

Then, in 2019, Hamer et al. [24] proposed a unique self-sovereign identity management scheme to deal with the above problem. A user's biometrics was authenticated by the trusted organization to make sure that the user did have this biometrics. Besides, the collected biometrics were encrypted with a homomorphic signature algorithm to ensure that a user could not enroll twice in the system. But all the user's behaviors can be linked to the same digital identity, which leaks the user's privacy.

A blockchain-based privacy protection unified identity authentication scheme is proposed in [23]. In this scheme, the server would authenticate user information by online face verification using photos from a central database. In addition, a set of key derivation algorithms were designed to ensure the unlinkability between identity attribute information. However, the way a central database stores user information weakens users' control over their identities.

In 2021, Bandara et al. [26] proposed a blockchain and self-sovereign identity empowered digital identity platform. For full control over the data, the information required for registration (name, address, photo, etc.) submitted by the user was stored locally on the mobile device. And only with the user's consent, these information would be sent to the service provider (SP). Additionally, verification of information is achieved by comparison with physical documents, eliminating the need for SPs to store information.

In a word, strict identity authentication, especially the introduction of biometrics, ensures that each user has one and only digital identity, effectively resisting sybil attacks. However, none of these schemes design accountability mechanisms to regulate malicious users who disrupt the order of the network.

2.3. SSI Schemes with Accountability. For the purpose of maintaining order in cyberspace, SSI schemes with accountability [27, 28] are proposed. In 2021, Stokkink et al. [27] designed a truly self-sovereign identity system based on Pedersen commitments, where the digital identities were implemented as data structures that held a list of credentials. And, for self-sovereignty, these data structures were stored on the users' devices. In terms of accountability, credential verifiers were required to keep audit logs, which were actually composed of credentials presented by users. Then, malicious users could be identified by a single regulatory authority through analyzing audit logs and comparing them with the personal information in a central registry. But several problems such as inadequate regulation and injustice may arise due to the reliance on a single regulatory authority.

Also in 2021, Maram et al. [28] presented a decentralized System model. Identity management with legacy compatibility, sybil-resistance, and accountability. Instead of an additional credential issuer, all credentials characterizing the user's identity in this scheme were imported from existing web service providers. And a deduplication protocol based on secure multiparty computation (MPC) was designed to prevent the reuse of these credentials. Besides, in order to address the drawbacks of the single regulatory authority, a MPC-based fuzzy matching method was proposed, which can find the digital identities of the corresponding malicious users according to the sanctions lists. However, legacy compatibility does not change the status quo of data stored by existing web service providers, which remains out of the user's control. In addition, both the central registry and the sanctions lists introduced in the above schemes to regulate malicious behavior inevitably sacrifice users privacy.

3. Models and Design Goals

3.1. System Model. Our system model consists of seven entities, as Figure 1 shows, a natural person (NP), a digital avatar-i (DA), two blockchains: an identity information chain (IIC), a digital avatar-i behavior chain (DABC), and three groups: an information collection and verification group (ICVG), a digital avatar-i generation group (DAGG), and a regulatory authority group (RAG).

(1) **NP**, Natural Person, refers to a person living in the physical world. He/She can digitize himself/herself through the ICVG and apply to the DAGG for a DA.

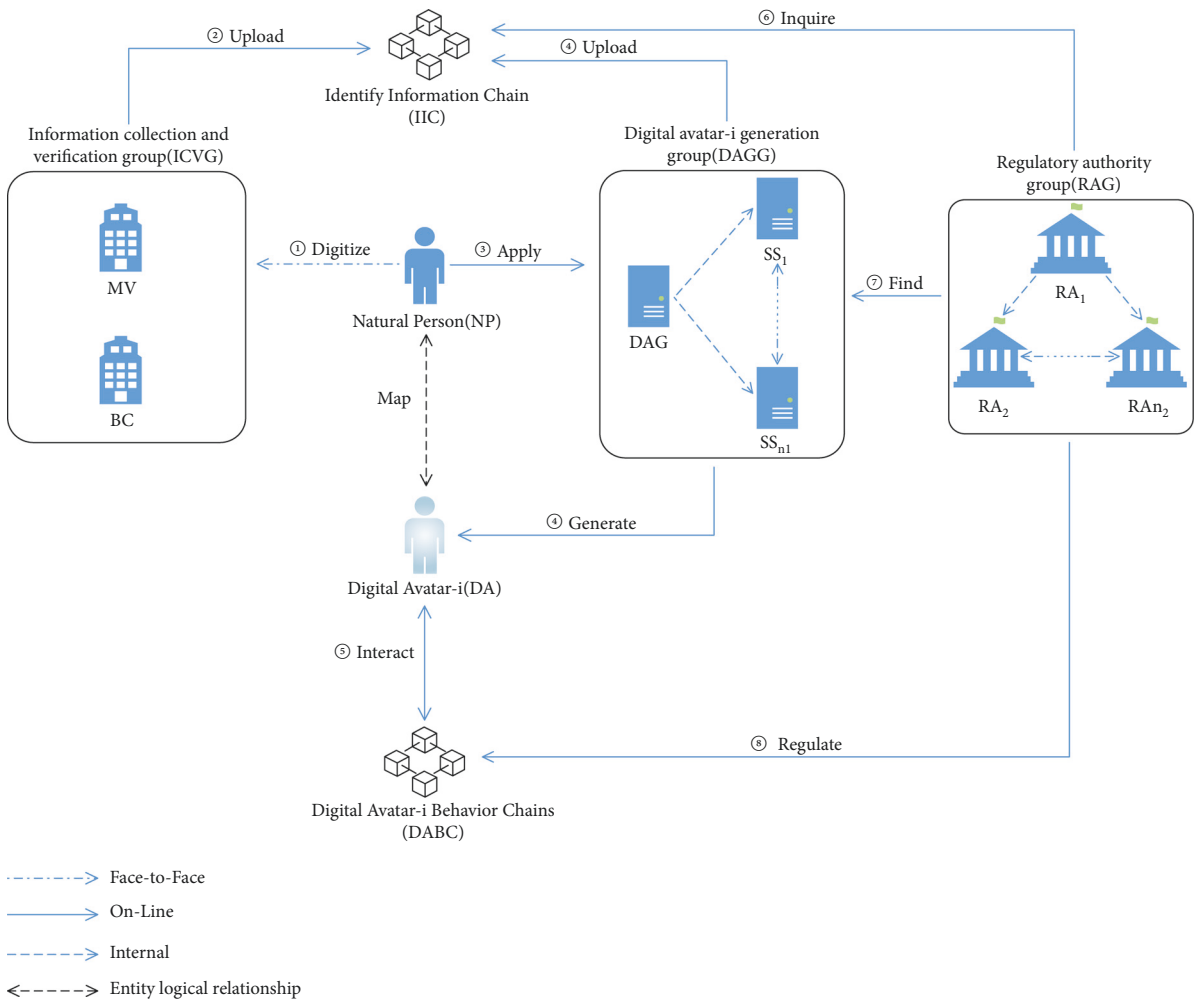


Figure 1: System model.

(2) **DA**, Digital Avatar-i, is an individual-specific executable program focusing on the identity dimension of the digital avatar, which stands for a living person to interact with others in cyberspace, and has one-to-one relationship with NP.

(3) **ICVG**, Information Collection and Verification Group, validates that the requestor is one and only breathing person in the physical world and provides digitalizing service for him/her. It contains two types of entities, namely, metadata verifier (MV) and biometric collector (BC). MV proves the requestor's existence in physical space through metadata, such as name, identity number, and address. BC collects two types of distinct biometric data, where one is as a permanent proof and the other is for activating the DA.

(4) **IIC**, Identity Information Chain, is a consortium blockchain. It is mainly responsible for recording the proof of physical identity information (metadata and biometric data), the hash of DA, and making sure each NP has only one proof.

(5) **DAGG**, Digital Avatar-i Generation Group, generates a unique DA for each NP. It contains two types of entities, namely, digital avatar-i generator (DAG) and secure storages (SSs). At first, DAG verifies the identity of the applicant with the data in the IIC, and then generates the sole DA for him/her. SSs, which contain $SS_1 \cdots SS_n$, use shamir(t, n) threshold algorithm to safely store the metadata of NP and the hash of DA.

(6) **DABC**, Digital Avatar-i Behavior Chains, is an infrastructure that is composed of multiple blockchains, supporting all kinds of decentralized applications (Dapps). These Dapps provide services for DAs in cyberspace and the DABC keeps their historical records for accountability.

(7) **RAG**, Regulatory Authority Group, is responsible for regulating NP by monitoring the DA's activities in the DABC. And it is composed of n regulatory authorities ($RA1 \ldots RA_n$), where at least three of them can hold suspicious users accountable.

In order to securely and privately take part in various activities such as work, study, and entertainment in cyberspace, especially the metaverse, a NP needs to map himself/herself to one DA. In detail, there are four steps to achieve the mapping. First, the NP needs to digitize himself/

herself through the ICVG, where MV verifies the descriptive metadata of the NP and BC collects the NP's biometric data. Second, the ICVG uploads the proof information to the IIC and replies the NP with a certificate. Third, the NP applies to the DAGG for a DA with the certificate. At last, the DAGG verifies the authenticity of the NP by checking whether the live biometric data matches the proof in the IIC. If the verification is passed, the DAGG generates the DA and uploads the hash of the DA to the IIC. Afterward, the NP uses the corresponding DA to live in the cyberspace without a third-party program.

It is worth noting that, once there is a malicious DA, the RAG can map him/her to the corresponding NP through inquiring the IIC and finding the metadata in DAGG. For constructing a safe and orderly cyberspace, a DA is supposed to interact with the Dapps based on DABC. In this way, the RAG can regulate a malicious NP through monitoring the DA's historical and future behaviors.

3.2. Security Model. In NSSIA, we have the following security assumptions.

(1) An adversary can monitor, intercept, modify, and insert the messages into the public channel [34]. He/she can breach no more than half of the entities in each group of ICVG, DAGG, and RAG within a certain period of time.

(2) A NP and a DA are considered as malicious entities. A NP would submit false information or fraudulently use anyone's information, and a DA can be modified or illegally used by an adversary in the cyberspace.

(3) Entities in the ICVG, DAGG, and RAG are regarded as semihonest. They will perform the protocol strictly and comply with the consensus algorithm of the IIC but are curious about the information.

(4) The DABC is a semi-honest entity. It is a public chain that is secured by consensus algorithms, and the miners perform the protocol strictly but are curious about the information. The IIC is a trusted consortium chain that is jointly managed by the ICVG, DAGG, and RAG.

(5) We assume that the standard cryptographic algorithm used in our scheme is secure and unbreakable.

3.3. Design Goals. According to the aforementioned system model and security model, the design goals of our scheme are as follows:

(i) User friendly: a user (NP) accesses a service in cyberspace with a digital avatar-i (DA) in a convenient way and the user owns and controls it.

(ii) One-to-one: in order to build an orderly cyberspace, a user (NP) has one and only digital avatar-i (DA). And all the behaviors of the DA belong to the only one NP.

(iii) Linkability with condition: for the security of cyberspace and the privacy of a user (NP), the identity mapping (the NP and the DA) is encrypted and stored in a distributed way (different pieces of it in each SS_i). Only three or more of the RAs can jointly decrypt it and recover the detail of identity.

4. Proposed NSSIA

The NSSIA generates a unique digital avatar-i for a user to ensure his/her conditional identity privacy when interacting with each other in cyberspace, especially the metaverse. Concretely, the NSSIA is mainly divided into five phases. The first phase initializes the entities in the IIC, ICVG, DAGG, and RAG to generate their keys. The second phase lets a NP digitize himself/herself through the ICVG, where the ICVG verifies the authenticity of NP's metadata, collects NP's biometric data, and writes the proof information to the IIC, as shown in steps ①-② of Figure 1. And in the third phase, the NP applies to the DAGG for a DA in which the DAGG checks the metadata of the NP, generates a DA and records the DA generation transaction to the IIC, as shown in steps ③-④ of Figure 1. The NP can use his/her own DA to interact with the Dapps built on DABC in the fourth phase, as shown in step ⑤ of Figure 1. Lastly, the RAG regulates a malicious NP with the mapping DA's behaviors in the DABC and the data in the IIC and the DAGG, as shown in steps ⑥-⑧ of Figure 1. To elaborate the NSSIA clearly, we give the notations used in our scheme in Table 1.

4.1. Initialization. The IIC performs initialization to generate the public parameters, the master key (MK), and the corresponding subkeys (SubKs). In addition, the entities in RAG, DAGG, and ICVG generate their public and private keys.

4.1.1. IIC Initialization. The IIC performs initialization to generate the public parameters, the MK and the $\mathrm{Sub}K_e$, where the $\mathrm{Sub}K_e$ is the subkey of the entity e. In detail, it first selects a large prime p, an elliptic curve $E_p(a,b)$ and a base point G with order n under the finite field F_p. Then, it publishes the public parameters $P = \{p, E_p(a,b), G, n\}$ to the genesis block. Afterward, it randomly selects a 128-bit AES key as the MK. Lastly, the IIC uses the shamir (t,n) threshold secret sharing algorithm [35] to generate the $\mathrm{Sub}K_e$ for each entities of DAGG and RAG. And we assume that the number of SSs and RAs is n_1 and n_2, and the corresponding thresholds are t_1 and t_2. Here are the details below.

The IIC first chooses two polynomials of degree $t_1 - 1$ and $t_2 - 1$ shown as (1) and (2), where $a_1, \ldots, a_{t_1-1}$, $b_1, \ldots, b_{t_2-1}$ are random numbers, and $N1$, $N2$ are bigger than each coefficient.

$$F_{SS}(x) = MK + a_1 x + \cdots + a_{t_1-1} x^{t_1-1} \mod (N1), \quad (1)$$

$$F_{RA}(x) = MK + b_1 x + \cdots + b_{t_2-1} x^{t_2-1} \mod (N2). \quad (2)$$

Table 1: Notations used in our scheme.

Notations	Description
MK	Master key
PK_e	Public key of entity e
SK_e	Secret key of entity e
$SubK_e$	Subkey of entity e
$ESubK_e$	Encrypted subkey of entity e
n_1, t_1	The number of secure storages is n_1 and the corresponding threshold is t_1
n_2, t_2	The number of regulatory authorities is n_2 and the corresponding threshold is t_2
$a \oplus b$	XOR operation of a and b
$H(\bullet)$	Hash operation on $\bullet$
$En(a,b)$	Use b to encrypt a
$De(a,b)$	Use b to decrypt a
$Sig(a,b)$	Use b to sign a
$Ver(a,b)$	Use b to verify a
SecInfo	Encrypted identity information

Next, the IIC chooses random numbers $x_i, i = 1, 2, \ldots, n_1 + n_2$ and substitutes the x_i into (1) and (2) to calculate the n_1 $SubK_{SS}s$ and n_2 $\mathrm{Sub}K_{RA}s$ $(n_k = 2 \times t_k - 1, k = 1, 2)$.

4.1.2. SS and RA Initialization. The SS and RA use the P published by IIC to generate their own public and private keys, referred to as PK_{SS}/SK_{SS} and PK_{RA}/SK_{RA}, and publish the public keys. Then, the encrypted subkeys with PK_{SS} and PK_{RA} are obtained from the IIC by the SS and RA. Next, they decrypt the encrypted subkeys to recover the $\mathrm{Sub}K_{SS}$ and $\mathrm{Sub}K_{RA}$.

4.1.3. MV, BC, and DAG Initialization. MV, BC, and DAG use the P published by IIC to generate their own public and private keys, referred to as PK_{MV}/SK_{MV}, PK_{BC}/SK_{BC}, and PK_{DAG}/SK_{DAG}.

4.2. Digitization. To prepare for the generation of a DA, the NP sends the metadata to the ICVG for digitizing himself/herself. Here, the MV verifies the metadata and records the proof of metadata to the IIC. While the BC collects the NP's biometric data, writes the proof of the biometric data to the IIC, and sends the NP a digitization credential. The whole process is shown in Figure 2.

STEP D1A NP presents the certificates, such as ID card and passport, and provides the metadata (MD_{NP}, including name, id number, address and gender) to the MV face to face, as shown in step ①.

STEP D2The MV verifies the authenticity of the MD_{NP} with the certificates. If it is confirmed, the MV calculates the proof $H_M = H(MD_{NP})$ and sends a metadata verification transaction (TM, as shown in Equation (3)) to the IIC, without any information (neither MD_{NP} nor H_M) stored locally, as shown in step ②.

$$TM = (Ti\ d, Tin[PK_{MV}, \phi, \phi], Tout[PK_{BC}, H_M, \omega]). \tag{3}$$

In the (3), according to the paper [36], Tid represents the transaction number of the TM. The input array Tin [] consists of three parts, the input address, the previous transaction, and the input script. The PK_{MV}, the input address, is the initiator's public key, since TM is the original transaction, both the last transaction of the TM and the input script are denoted to the Φ. The output array Tout[] is composed of three parts, where the PK_{BC} is the accepter's address, H_M is the data to be recorded in the IIC, and ω is an out-script used to sign the TM.

STEP D3The MV sends the transaction number (TNum) of TM to the NP, as shown in step ③.

STEP D4The NP sends the MD_{NP}, TNum to the BC face to face for authentication, as shown in step ④.

STEP D5The BC calculates $H_M^{'} = H(MD_{NP})$, uses TNum to find the H_M of TM recorded in the IIC, and checks whether $H_M^{'} = H_M$ is satisfied. If not, the BC aborts and it is shown in step ⑤.

STEP D6The BC collects two kinds of biological characteristics, where the one is as a permanent proof of NP's existence in cyberspace and the other one is used to activate the DA. Specifically, the permanent one should be unbreakable and needs not to be collected frequently, therefore, we choose the iris data. While for frequently using the DA to access network services, an easy-to-collect face biometric is introduced. And then, the BC calculates the $H_I = H(iris)$ and sends a iris verification transaction (TI, as shown in (4)) to the IIC, with no information (such as biometrics and H_I) stored locally, as shown in step ⑥.

$$TI = (Ti\ d, Tin[PK_{BC}, TM, \varphi], Tout[PK_{DAG}, H_I, \omega]). \tag{4}$$

In the (4), the Tid is the transaction number of the TI, the PK_{BC} is the creater's address of the TI, the TM is the previous transaction, the PK_{DAG} is the accepter's address of the TI, and the H_I is the data to be recorded in the IIC.

STEP D7As Equation (5) shows, the BC encrypts the face data with the PK_{DAG} and then signs it with the SK_{BC} to generate the $M1$.

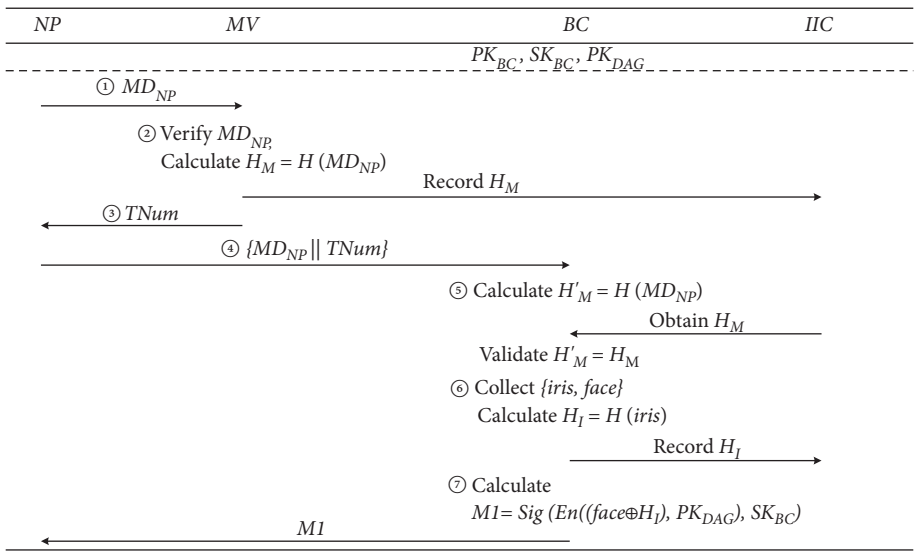


Figure 2: The flow chart of digitizing a NP.

$$M1 = Sig\left(En\left((\text{face}\oplus H_I), PK_{DAG}\right), SK_{BC}\right). \quad (5)$$

Finally, the BC sends $M1$ to the NP, as shown in step ⑦.

4.3. Generation. After the digitization, the DAGG can generate a DA for the NP and the process consists of seven steps. At first, the NP applies to the DAGG for a DA. Second, the DAG verifies the authenticity of NP's identity with the proof information in the IIC. Third, the DAG generates a DA and requests the SSs' subkeys. Then, the SSs send the encrypted subkeys to the DAG. Next, the DAG restores the MK to generate the $SecInfo$, and splits it into multiple backup information. Afterward, the DAG records the proof of DA in the IIC, and lastly sends the DA to the NP, as shown in Figure 3.

STEP G1The NP sends the physical identity proof $PIP = \{MD_{NP}, TNum, M1\}$ to the DAG, as shown in step ①.

STEP G2The DAG calculates $H''_M = H(MD_{NP})$ and M2 by Equation (6).

$$M2 = De\left(Ver\left(M1, PK_{BC}\right), SK_{DAG}\right). \quad (6)$$

Afterward, the DAG obtains the H_M and the H_I with TNum from the IIC and checks whether $H''_M = H_M$ is met. At last, the DAG calculates $M3 = M2\oplus H_I$, and verifies the living face biometric of NP with the $M3$, as shown in step ②.

STEP G3If the NP's identity is confirmed, according to Algorithm 1, the DAG selects corresponding code modules (dynamic verification, file transfer, etc.) from the code library to get the DA with the digital avatar-i seed (DAS) which is produced from $M3$ by the algorithm in the paper of [37]. The DA is divided into k modules and the selected code modules are combined together in order. Then, the DAG calculates the identifier $DA\ I = H(DA)$ and requests all SSs to send their respective $SubK_{SS}$, as shown in step ③.

STEP G4Each SS_i encrypts subkey to get $ESubK_{SS_i}$ by the Equation (7) and sends it to the DAG, as shown in step ④.

$$ESubK_{SS_i} = En\left(SubK_{SSi}, PK_{DAG}\right). \quad (7)$$

STEP G5The DAG calculates at least $t_1 - 1 SubK_{SS_i} = De\left(ESubK_{SS_i}, SK_{DAG}\right), i = 1, 2, \ldots, t_1 - 1$ and constructs the Lagrangian interpolation formula (as shown in Equation (8)) with these $SubK_{SS_i}s$ to restore the $MK = f'(0)$.

$$f'(x) = \sum_{i=1}^{t_1} y_i \prod_{j=1, j\neq i}^{t_1} \frac{\left(x - x_j\right)}{x_i - x_j}. \quad (8)$$

Afterward, the DAG generates the $SecInfo = En((MD_{NP}\oplus DA\ I), MK)$ and expands SecInfo to $n \times t_1 \times b$ bytes by filling high bits with zero. The DAG constructs n polynomials, as shown in (9).

$$\begin{cases} F_1(x) = m_0 + m_1 x + \cdots + m_{t_1-1}x^{t_1-1} \mod(N), \\ F_2(x) = m_{t_1} + m_{t_1+1}x + \cdots + m_{2t_1-1}x^{t_1-1} \mod(N), \\ F_n(x) = m_{(n-1)t_1} + m_{(n-1)t_1+1}x + \cdots + m_{nt_1-1}x^{t_1-1} \mod(N). \end{cases} \quad (9)$$

In the (9), the SecInfo is divided into $n \times t_1$ coefficients in order and each coefficient is b bytes. The N is a prime number bigger than any coefficient m_i, and the length of N is $b + 1$ bytes.

The DAG substitutes n_1 xs into each polynomial in the (9) to calculate $n \times n_1$ points $(x_i, y_{ij}), i = 1, 2, \ldots n_1, j = 1, 2, \ldots, n$ (since x_i performs multiple exponentiation operations, the length of x_i is set to one byte to reduce computational overhead, while the length of y_{ij} is set to five bytes to avoid collisions between these points. If the lengths of x_i and y_{ij} are too short, the high bits are filled with zero). Further, the DAG divides all the points into

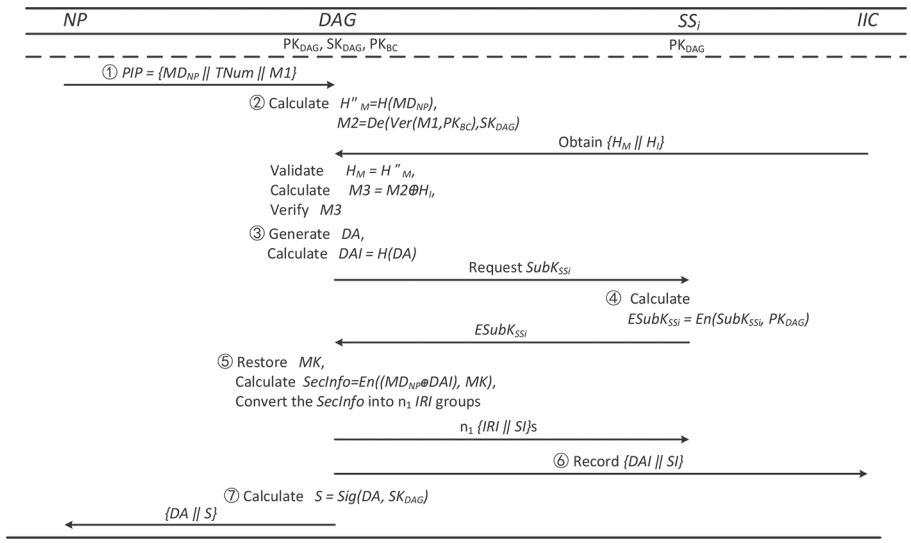


Figure 3: The flow chart of generating a DA.

n_1 groups and n points in each group come from the different $F(x)s$. It is worth mentioning that the x_i in these n points are the same, and the n points (x_i, y_{ij}) are combined to form a set of identity restoration information (IRI), which is as shown in Figure 4. At last, the DAG transmits n_1 sets of IRI and storage index (SI) to all the SSs, as shown in step ⑤.

STEP G6. If the SS_i receives the IRI and SI, he/she sends a response to the DAG. When DAG confirms that more than half of SSs have received IRI and SI, he/she writes a DA generation transaction (TDA, as shown in (10)) in the IIC, as shown in step ⑥.

$$\begin{aligned} T\ DA = (&Ti\ d, Tin[PK_{DAG}, TI, \varphi], \\ &Tout[PK_{DAG}, (DA\ I \| SI), \omega]). \end{aligned} \tag{10}$$

In the Equation (10), the Tid is the transaction number of the TDA, the TI is the previous transaction, the PK_{DAG} is the creater and the accepters' address of the TDA, and the $(DA\ I \| SI)$ is the data to be recorded in the IIC.

STEP G7The DAG calculates the DA's proof $S = Sig(DA\ , SK_{DAG})$ and sends it with the DA to the NP, as shown in step ⑦.

4.4. Interaction. After receiving the DA, the NP can access various services provided by Dapps built on DABC through it. At first, the NP activates the DA through live face recognition. And then, the DAI or a random string can be selected by the activated DA as the identifier for the NP to participate in activities in cyberspace. It is worth mentioning that all behaviors of the NP accessing network services will be recorded in the DABC for future audit.

In a word, the main work of this phase is to use DA for authentication and authorization, which requires unlinkable identity, informed consent and the right to be forgotten, etc. However, limited by space, details such as the protocol process, algorithms, and data format will be given in our future work.

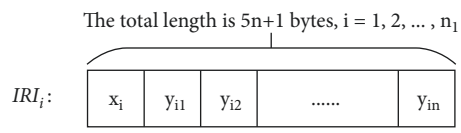


Figure 4: The format of the IRI_i.

4.5. Accountability. When a malicious behavior of a DA occurs, the RAG can discover the mapping NP by inquiring the IIC and finding the metadata of the NP in the DAGG with the joint participation of multiple RAs. Then, the RAG can regulate all the historical behaviors of the malicious NP, as shown in Figure 5.

STEP S1All the RAs are monitoring the DAs' behavior in DABCs, as shown in step ①.

STEP S2When the RA_i finds a suspicious behavior of a DA, he/she can start the accountability mechanism. First, RA_i inquires SI from the IIC with DAI and writes an audit transaction (TA, as shown in (11)) in the IIC, as shown in step ②.

$$\begin{aligned} TA = \big(&Ti\ d, Tin\big[PK_{RA_i}, TA, \phi\big], \\ &Tout\big[PK_{RA_i}, (\text{timestamp} \| DA\ I), \omega\big]\big). \end{aligned} \tag{11}$$

In the Equation (11), the Tid is the transaction number of the TA, the TA in the Tin[] is the previous audit transaction, the PK_{RA_i} is the creater and the accepters' address of this TA, and the $(\text{timestamp} \| DA\ I)$ is the data to be recorded in the IIC.

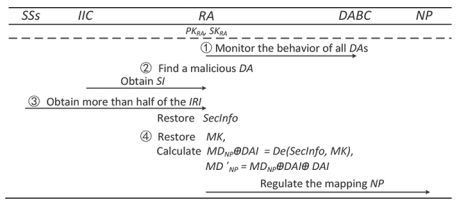


Figure 5: The flow chart of regulating the NP.

STEP S3Then, the RA_i finds at least t_1 IRIs stored in SSs with the SI. And all the obtained IRIs are processed as follows: ① The RA_i decomposes each IRI_j, $j = 1, 2, \ldots, t_1$ into n points, where the structure of the IRI_j is shown in Figure 4 and the decomposed point set is shown in Equation (12):

$$\begin{Bmatrix} (x_1, y_{1,1}) & (x_1, y_{1,2}) & \cdots & (x_1, y_{1,n}) \\ (x_2, y_{2,1}) & (x_2, y_{2,2}) & \cdots & (x_2, y_{2,n}) \\ \vdots & \vdots & \ddots & \vdots \\ (x_{t_1}, y_{t_1,1}) & (x_{t_1}, y_{t_1,2}) & \cdots & (x_{t_1}, y_{t_1,n}) \end{Bmatrix}. \tag{12}$$

② Then, substituting the t_1 points $(x_1, y_{1,1})$, $(x_2, y_{2,1})$, $\ldots$, $(x_{t_1}, y_{t_1,1})$ into the Lagrangian interpolation formula (13) to obtain the polynomial $F_1(x)$;

$$F'(x) = \sum_{i=1}^{t_1} y_i \prod_{j=1, j\neq i}^{t_1} \frac{(x - x_j)}{x_i - x_j}. \tag{13}$$

③ Similar to step ②, the RA_i obtains the polynomials $F_2(x), \cdots, F_n(x)$.

After the n polynomials are obtained, the $SecInfo$ is restored by splicing the coefficients of these polynomials in order, as shown in step ③.

STEP S4. The RA_i initializes an audit request to all other RA_js. Each RA_j encrypts his/her $SubK_{RA_j}$ by the (14) and sends the $ESubK_{RAj}$ to the RA_i.

$$\begin{aligned} ESubK_{RA_j} &= En\left(SubK_{RA_j}, PK_{RA_i}\right) \\ &\left(j = 1, 2, \ldots, n_2; j \neq i\right) \end{aligned}. \tag{14}$$

When more than t_2 RA_js respond, the RA_i decrypts the $ESubKey_{RA}$ one by one using the (15).

$$\begin{aligned} SubK_{RA_j} &= De\left(ESubK_{RA_j}, SK_{RA_i}\right) \\ &\left(j = 1, 2, \ldots, t_2; j \neq i\right) \end{aligned}. \tag{15}$$

Then, the RA_i constructs the Lagrangian interpolation formula with the t_2 $SubK_{RA}$s, as shown in the (16), to calculate the $MK = f'(0)$.

$$f'(x) = \sum_{i=1}^{\lfloor t_2 \rfloor} y_i \prod_{j=1, j\neq i}^{\lfloor t_2 \rfloor} \frac{(x - x_j)}{x_i - x_j}. \tag{16}$$

After that, the RA_i decrypts the $SecInfo$ to get the $(MD_{NP} \oplus DA\ I) = De(SecInfo, MK)$, and gets $MD_{NP}' = (MD_{NP} \oplus DA\ I) \oplus DA\ I$. So far, the RA_i can discover the malicious NP and regulate him/her through the historical behaviors in the DABC, as shown in step ④.

5. Security Analysis

In this section, we discuss the security of the proposed scheme.

5.1. Conditional Anonymity. In this scheme, the metadata of a NP (MD_{NP}) is hidden in the SecInfo by the $DA\ I$ and MK, and the other entities except RA cannot recover it without the $DA\ I$ and MK. The $DA\ I$ is recorded on the IIC, while the shamir (t, n) threshold algorithm protects the MK. Therefore, the scheme realizes the anonymity of entities other than the RA. On the other hand, we allow at least t RAs to restore the MK for revealing the MD_{NP} by the Lagrangian interpolation formula. In short, the conditional anonymity is achieved in our scheme.

5.2. Anti-Sybil Attack. In this scheme, each NP needs to digitize himself/herself through the ICVG before applying for a DA, where the authenticity of the NP's MD_{NP} is verified by the MV with NP's certificates and the NP's biometric data is collected by the BC as a proof of unique identity. In addition, the hash of the MD_{NP} and biometric data (iris) are permanently recorded on the IIC. In this way, it can be ensured that each NP has one and only DA, and the sybil attack is avoided.

5.3. Tamper-Proof. During the digitization phase, the NP is required to provide MD_{NP} face-to-face, therefore, the tampered MD_{NP} submitted by the adversary cannot be verified by the MV with the NP's certificates. The consortium blockchain records the proof of MD_{NP} and biometric data, no one can easily erase the NP's information. In addition, the DAG calculates the signature $S = Sig(DA\ , SK_{DAG})$ to prevent the adversary to tamper with the DA.

5.4. Nonrepudiation. For each DA, the RAG can find the corresponding SecInfo through the data in the IIC and SSs. Then, the RAG can calculate the MK with the participation of multiple RAs, and get $MD_{NP} \oplus DA\ I = De(\text{SecInfo}, MK)$. Using the known $DA\ I$ in the IIC, the RAG can obtain the $MD_{NP} = MD_{NP} \oplus DA\ I \oplus DA\ I$, and track the mapping NP with it. That is, a NP cannot deny his/her malicious historical behavior.

5.5. Impersonation-Resistance. When accessing a DA, the NP's biometric data needs to be verified by the DA in advance. In addition, the DA includes a dynamic verification module that will issue dynamic verification requests to the NP from time to time. Once the NP fails to pass the verification, the DA will be locked. Therefore, even if an adversary obtains a DA that does not belong to him/her, he/she cannot use it to participate in network activities.

5.6. Data Security. The metadata of a NP (MD_{NP}) and the hash of a DA ($DA\ I$) are first encrypted as the SecInfo, then divided into $n \times t_1$ parts, and finally converted into $n \times n_1$ points by shamir (t, n) threshold algorithm. In this way, the cost of obtaining a NP's information by an adversary is greatly increased. Further, the information transmitted between different entities, such as subkeys, are protected by asymmetric encryption algorithm. Therefore, the scheme guarantees the security of the data.

5.7. Provable Security. The proposed scheme is based on the advanced encryption standard (denoted as AES), elliptic curve cryptography (denoted as ECC), and shamir (t, n) threshold algorithm (denoted as SHAMIR). According to the security characteristics of each module, we show that our scheme can resist sybil attacks of adversaries and prevent malicious users from evading sanctions.

Theorem 1. If the ECC algorithm satisfies the basic security properties, then the scheme in this paper can meet sybil-resistant characteristics.

Proof 1. Define A_{ECC} as an adversary attacking the security of ECC algorithm. Assuming A_{SR} successfully carried out a sybil attack, a polynomial time algorithm $A_\theta \in (A_{ECC})$ is defined, where the A_{SR} has the ability to attack the ECC algorithm. Through the interaction between A_{SR} and A_θ in the simulated sybil attack game, A_θ is optimized repeatedly to successfully attack the ECC algorithm. That is, if the adversary A_{SR} creates a sybil identity successfully in this scheme, it means A_θ successfully attack the security of ECC algorithm with a certain probability. According to the steps defined above, the interactions between the algorithm A_θ and adversary A_{SR} are as follows:

STEP 1**Initialization phase:** Through the public parameters generated by A_{ECC} in IIC Initialization phase, the algorithm A_θ generates the PK_{MV}, PK_{BC}, and $PK_{DA\ G}$, and sends them to the adversary A_{SR};

STEP 2**Challenge phase:**The adversary A_{SR} first digitizes the identity information (including metadata and biometrics) as described in Section 4.2. Then, the algorithm A_θ executes A_{ECC} algorithm to calculate the encrypted biometrics containing a randomly chosen $b \in \{0, 1\}$, and generate the corresponding signature $M1$. Finally, A_θ sends the signature $M1$ to the adversary A_{SR}.

STEP 3**Verification phase:** The adversary A_{SR} verifies the signature $M1$ with the A_{ECC} and outputs the b'. If the equation $b' = b$ is satisfied, it indicates that the adversary A_{SR} successfully implemented the sybil attack. The probability of success for the adversary A_{SR} is:

$$\begin{aligned} Adv_{A_{SR}}(k) &= \Pr\left[Exp_{A_{SR}}(k) = 1\right] \\ &= \Pr\left[A_{SR}(\text{verify}) = 1 | b = 1\right] \cdot \Pr[b = 1] \\ &\quad + \Pr\left[A_{SR}(\text{verify}) = 0 | b = 0\right] \cdot \Pr[b = 0] \\ &= \frac{1}{2}\Pr\left[A_{ECC}(\text{verify}) = 1 | b = 1\right] \\ &\quad + \frac{1}{2}\Pr\left[A_{ECC}(\text{verify}) = 0 | b = 0\right] \\ &= \Pr\left[Exp_{A_{ECC}}(k) = 1\right] \\ &= Adv_{A_{ECC}}(k) \end{aligned} \quad . \tag{17}$$

If an attacker A_{ECC} can successfully attack ECC algorithm, A_{SR} can carry out the sybil attack successfully. However, the probability of A_{ECC} successfully attacking the ECC algorithm is almost $1/n$, then A_{SR} wins in the sybil attack game of NSSIA scheme with a probability of $1/n$. But, according to the assumption about the security of the ECC algorithm, the probability of A_{SR} successfully attacking can be ignored. Therefore, the scheme can resist sybil attack.

Theorem 2. If the AES and SHAMIR algorithms satisfy the basic security features, then the NSSIA can prevent malicious users from evading sanctions.

Proof 2. Define A_{AES} as an adversary who attacks the security of AES algorithm, A_{SHAMIR} as an adversary attacking the security of SHAMIR algorithm. Assuming the A_{JA} successfully hampered joint regulation of the malicious NP, a polynomial time algorithm $A_\psi \in (A_{AES}, A_{\text{SHAMIR}})$ is defined, where the A_{JA} has the ability to attack the algorithms of AES and SHAMIR. Through the query of A_{JA} and A_ψ's interaction in the sanctions evasion game, A_ψ is optimized repeatedly to successfully attack the AES and SHAMIR algorithms. That is, if the adversary A_{JA} gets rid of sanctions successfully in the scheme, it means A_ψ successfully attacks the security of algorithms of AES and SHAMIR with a certain probability. According to the steps defined above, the interactions between the algorithm A_ψ and adversary A_{JA} are as follows:

STEP 1**Initialization phase:** Through the public parameters generated by A_{AES} and A_{SHAMIR} in IIC Initialization phase, the algorithm A_ψ generates the master key (MK) and subkeys ($SubK_{SS}$, $SubK_{RA}$). Then, A_ψ sends the public parameters to the adversary A_{JA};

STEP 2**Inquiry phase:** The adversary A_{JA} can query the algorithm A_ψ for polynomial time:

(1) Generate the encrypted identity information SecInfo: A_ψ generates the encrypted identity information $SecInfo$ which contains a randomly selected $b \in \{0, 1\}$ by the AES algorithm.

(2) Generate multiple identity restoration informations IRI s: Based on the $SecInfo$, A_{ψ} generates multiple identity restoration informations $IRIs$ by the SHAMIR algorithm, and sends these $IRIs$ to the adversary A_{JA}.

STEP 3**Verification phase:** The adversary A_{JA} decrypts the $SecInfo$ restored by $IRIs$ and outputs the b' using the AES and SHAMIR algorithms. If $b' = b$ exists, it represents that the adversary A_{JA} successfully evades sanctions. The probability of success for the adversary A_{JA} is:

$$\begin{aligned}
A\ dv_{A_{JA}}(k) &= \Pr\left[Extp_{A_{JA}}n(k)q = h1\right] \\
&= \Pr\left[A_{JA}(verify) = 1|b = 1\right] \cdot \Pr[b = 1] \\
&\quad + \Pr\left[A_{JA}(verify) = 0|b = 0\right] \cdot \Pr[b = 0], \\
&= \frac{1}{2}\Pr\begin{bmatrix} A_{AES}(verify) = 1 \\ \\ A_{SHAMIR}(verify) = 1 \end{bmatrix} |b = 1 \\
&\quad + \frac{1}{2}\Pr\begin{bmatrix} A_{AES}(verify) = 0 \\ \\ A_{SHAMIR}(verify) = 0 \end{bmatrix} |b = 0, \\
&< \frac{1}{2}\left(\Pr\left[A_{AES}(verify) = 1|b = 1\right]\right. \\
&\quad + \Pr\left[A_{\mathrm{SHAMIR}}(verify) = 1|b = 1\right] \\
&\quad + \frac{1}{2}\left(\Pr\left[A_{AES}(verify) = 0|b = 0\right]\right. \\
&\quad + \Pr\left[A_{\mathrm{SHAMIR}}(verify)t = n0q|b = 0\right], \\
&= \Pr\left[Exp_{A_{AES}}(k) = 1\right] + \Pr\left[Exp_{A_{SHAMIR}}(k) = 1\right] \\
&= A\ dv_{A_{AES}}(k) + A\ dv_{A_{SHAMIR}}(k).
\end{aligned} \tag{18}$$

If an attacker A_{AES} successfully attacks the AES algorithm, and an attacker A_{SHAMIR} can successfully attack the SHAMIR algorithm, the adversary A_{JA} can successfully hamper joint regulation of the malicious NP. However, the probability of A_{AES} and A_{SHAMIR} successfully attacking the AES and SHAMIR algorithms is almost $1/n$ respectively, then A_{JA} wins in the sanctions evasion game of NSSIA scheme with a probability of $2/n$. But, according to the assumptions that AES algorithm and SHAMIR algorithm satisfy the basic security properties, it is concluded that the probability of A_{JA} successfully attacking can be ignored. As a result, the scheme can prevent malicious users from evading sanctions through joint accountability.

6. Performance Analysis

This section analyzes the cost of our proposed NSSIA and compares it with the above schemes [17, 22, 23, 26, 28] in terms of the SSI property in identity generation, computation cost, storage cost, and blockchain Gas cost.

6.1. Property Analysis in Identity Generation. Ten principles are proposed by Christopher Allen [7] to define a SSI model, which are Existence, Control, Access, Transparency, Persistence, Portability, Interoperability, Consent, Minimalization, and Protection. It can be said that Allen's insights on SSI lays the foundation for the research of later generations.

As the shortcomings of the centralized identity model have been revealed in recent years, more and more scholars have invested in the research of SSI, and their work can be seen from literature [1, 38]. It is worth mentioning that before this article is written, Ferdous et al. [38] had introduced in detail the insights of various scholars on SSI. At the same time, they put forward their own views on the properties of SSI. They divided self-sovereign identity into five categories, with a total of seventeen properties. Mühle et al. [1] analyzed the work of Christopher Allen and then studied four basic components for having a deeper understanding of the concept of SSI.

As the identity generation is the crucial step for users to enter cyberspace, in order to effectively guarantee the users' self-sovereignty over identities and maintain the order of the cyberspace, we believe that the first step in the generation phase is to ensure that each DA has a corresponding NP to avoid false identities. In addition, it is also crucial to ensure that NPs fully control their own DAs and protect the privacy of users. Finally, DA should be user friendly, e.g., while complying with regulations, DA should be used for as long as possible and NP can migrate DA-related data between different devices. Therefore, we select the applicable six properties among the seventeen properties proposed by Ferdous et al. [38], as shown in Figure 6, and conduct the analysis. These properties are depicted next.

(1) **Existence.** Digital identities should be strictly verified before registration to ensure that each digital identity has a corresponding physical entity.

(2) **Ownership.** Digital identities can only be held and controlled by users.

(3) **Protection.** The registration of digital identities should pay attention to protecting user privacy and avoiding the identity link between the physical world and the cyberspace. At the same time, the design of SSI model should prevent fraudulent use of identity by others.

(4) **Persistence.** The digital identity should exist forever if the user does not take the initiative to revoke.

(5) **Portability.** When the user's device is replaced or the system's infrastructure is updated, the user's data can be easily transferred to the new device.

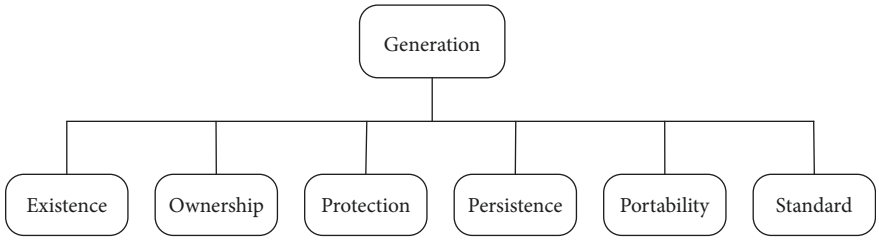


Figure 6: Taxonomy of the identity generation.

Input: The digital avatar-i seed DAS The code module template CMT_i, $1 \le i \le k$
Output: The digital avatar-i DA
(1) $len = strlen(DAS)/k$;
(2) $DA = 0$, $m = 0$, $n = len$;
(3) **For** $i = 0; i \le k - 1; i++$ **do**
(4) $str = DAS.substring(m + len \times i, n + len \times i)$;
(5) $a[i] = De\ cimal(str) \mod (num[i])$;
//Decimal is a function that converts a string to
//a decimal
//num[i] is the number of code module templates
//available in CMT_i
(6) $DA = Combine(DA, CMT_i(a[i]))$;
//Combine is a function that splices code
//modules in order.
(7) **End for**
(8) **return** DA;

Algorithm 1: Digital Avatar-i Generation.

(6) **Standard.** The SSI model should comply with the laws and regulations of various countries and international standards, such as GDPR and DID.

Next, we compare our scheme with the previously mentioned schemes on these properties, as shown in Table 2.

From Table 2, the properties of "Ownership" and "Persistence" are satisfied in [17], and the part of "Protection", "Portability" and "Standard" properties are met; however, the "Existence" property is unsatisfied. In [22], "Ownership" and "Persistence" properties, as well as the part of "Protection" and "Standard" properties are fulfilled, while the "Existence" property is unsatisfied and the rest is uncertain. In [23], "Existence" and "Protection" properties as well as the part of "Ownership" property are satisfied, but the "Standard" property is unsatisfied and the others is doubtful. The properties of "Existence", "Ownership," and "Persistence" properties are satisfied in [26], and the part of "Protection," "Portability," and "Standard" is met. Maram et al. [28] fulfill the properties of "Ownership" and "Persistence," as well as the part of "Existence," "Protection" and "Standard" properties, but the rest is doubtful. Compared with them, these properties are all realized in our scheme.

6.2. Computation Cost

6.2.1. The NSSIA. Our scheme includes five phases, namely, initialization, digitization, generation, interaction, and accountability. Since the initialization phase is executed only once and the interaction phase is not the point, we do not evaluate these two phases, and we mainly focus on the other three phases. Our scheme is run on a PC with windows 10, Intel(R) Core(TM) i5-1035G1 CPU @ 1.00 GHz and RAM 16G. We use Java 8.0 and Python 3.9 to evaluate the computation cost, where we choose the SHA-1 algorithm, the AES-128 algorithm, and the Secp256k1 elliptic curve. In order to balance security and computational cost, the n_1 and n_2 which is the number of SSs and RAs, respectively, are set to 5 in this simulation, and the corresponding t_1 and t_2 are 3. In addition, the length of the $SubK_e$ is 17 B. The MD_{NP}, DA, n and b are 256 B, 1 KB, 20, and 5 B separately. That is, the length of $SecInfo$ is $t_1 \times n \times b = 300$ B. According to the data in the paper [39], the length of iris and face is 25 KB and 30 KB, respectively.

To elaborate the computation cost clearly, a series of computational notations are defined:

(1) T_{H1} denotes the SHA-1 operation. T_{H1_1}, T_{H1_2}, and T_{H1_3} are the computation cost of performing the SHA-1 operation when the parameter sizes are 256 B, 1 KB, and 25 KB, respectively, and they are 0.0169 ms, 0.0353 ms, 0.0748 ms.

(2) T_{S1} indicates the AES-128 encryption/decryption algorithm and the computation cost is 0.4186 ms when the parameter size is 256 B.

(3) T_{AE} represents the ECC encryption algorithm. T_{AE_1} and T_{AE_2} are the computation cost of performing the

Table 2: Property comparison.

Schemes	Existence	Ownership	Protection	Persistence	Portability	Standard
[17]	N	Y	P	Y	P	P
[22]	N	Y	P	Y	–	P
[23]	Y	P	Y	–	–	N
[26]	Y	Y	P	Y	P	P
[28]	P	Y	P	Y	–	P
ours	Y	Y	Y	Y	Y	Y

aThe "Y" and "N" symbols, respectively, indicate that a certain property is satisfied or not in the corresponding scheme. The "P" symbol indicates that part of the property is satisfied. The "–" symbol shows that there is no obvious information about whether the property is satisfied or not.

ECC encryption algorithm when the parameter sizes are 17 B and 30 KB, respectively, and they are 0.0015 ms and 0.0024 ms.

(4) $T_{A\ D}$ expresses the ECC decryption algorithm. T_{AD_1} and T_{AD_2} are the computation cost of performing the ECC decryption algorithm when the parameter sizes are 17 B and 30 KB, respectively, and they are 0.034 ms and 0.0401 ms.

(5) T_{Sig} serves as the ECDSA signature algorithm. T_{Sig_1} and T_{Sig_2} are the computation cost of performing the ECDSA signature algorithm when the parameter sizes are 1 KB and 30 KB, respectively, and they are 0.0014 ms and 0.0283 ms.

(6) T_{Ver} is the ECDSA verification algorithm. T_{Ver_1} and T_{Ver_2} are the computation cost of performing the ECDSA verification algorithm when the parameter sizes are 1 KB and 30 KB, respectively, and they are 0.0007 ms and 0.001 ms.

(7) T_L is the Lagrangian interpolation algorithm and the computation cost is 0.0101 ms when the threshold value is $t = 3$.

Since the time cost of XOR operation, split operation, and splicing operation is negligible, it is not taken into consideration.

The computation cost of different phases in our scheme is shown in Table 3.

In Table 3, three SHA-1 operations, one ECDSA signature algorithm, and one ECC encryption algorithm are performed in the Digitization phase and the time cost is $2T_{H1_1} + T_{H1_3} + T_{Sig_2} + T_{AE_2} = 0.1393$ ms. Two SHA-1 operations, two ECC encryption algorithms, one ECDSA verification algorithm, one ECDSA signature algorithm, one Lagrangian interpolation algorithm, one AES encryption, and three ECC decryption algorithms are performed in the Generation phase and the time cost is $T_{H1_1} + T_{H1_2} + 2T_{AE_1} + T_{Ver_2} + T_{Sig_1} + T_L + T_{S1} + 2T_{AD_1} + T_{AD_2} = 0.5944$ ms. The Accountability phase consists of two ECC encryption algorithms, two ECC decryption algorithms, twenty-one Lagrangian interpolation algorithms, and one AES decryption algorithm and the time cost is $2T_{AE_1} + 2T_{AD_1} + 21T_L + T_{S1} = 0.7017$ ms. To generate a unique DA for each NP, the total $0.1393 + 0.5944 + 0.7017 = 1.4354$ ms is used.

6.2.2. Computation Cost Comparison. We compare with other schemes in terms of the generation phase and the accountability phase, as shown in Table 4.

In Table 4, one PBKDF2 algorithm, one SHA-256 operation, and two AES-256 encryption algorithms are performed in the generation phase in [17], and the time cost is $T_P + T_{H2} + T_{S2_1} + T_{S2_2} = 40.5941$ ms. In [22], one SHA-1 operation, one ECDSA signature algorithm, and one zk-SNARK algorithm are performed in generation phase, in which the time cost is $T_{H1_1} + T_{Sig_1} + T_Z = 18217.9183$ ms. Zheng et al. [23] perform two AES-128 encryption algorithms, three ECC encryption algorithms, three ECC decryption algorithms, three ECDSA signature algorithms and three ECDSA verification algorithms in generation phase, and the time cost is $3T_{AE_2} + 3T_{Sig_1} + 3T_{Ver_1} + 2T_{S1} + 3T_{AD_2} = 0.971$ ms. In [26], one base58 encoding algorithm and two RSA signature algorithms are performed in generation phase, and the time cost is $T_B + 2T_R = 6.0892$ ms. One oracle operation and one ZKP operation are performed in generation phase in [28], and the time cost is $T_O + T_{ZKP} = 2350$ ms. As shown in Section 6.1, the computation cost is $T_{H1_1} + T_{H1_2} + 2T_{AE_1} + T_L + T_{Sig_1} + T_{Ver_2} + 2T_{AD_1} + T_{S1} + T_{AD_2} = 0.5944$ ms in generation phase in our scheme.

For the accountability phase, since the corresponding mechanism has not been designed in the paper [17, 22, 23, 26], the audit cost cannot be given. While in [28], one secure multiparty computation operation is performed in the accountability phase, where the audit cost is $T_M = 1501540$ ms. As shown in Section 6.1, the accountability cost in our scheme is $2T_{AE_1} + 2T_{AD_1} + 21T_L + T_{S1} = 0.7017$ ms. From Table 4, our scheme has the lowest time overhead in both the generation phase and the accountability phase.

6.3. Storage Cost. We compare the storage cost with other schemes from the perspective of users, servers, and blockchain. The comparison result is shown in Table 5.

As shown in Table 5, users in [17] need to store a master key and a corresponding derived key, as well as the encrypted personal identity information, which are 168 bytes. And users in [22] need to store a random value key for hash algorithm, and the storage cost is estimated to be 16 – 32 bytes. In [23], three pairs of public and private keys, as well as a password used for symmetric encryption

Table 3: Computation cost on different phases.

Phases	Computation cost (ms)
Digitization	$2T_{H1_1} + T_{H1_3} + T_{Sig_2} + T_{AE_2} = 0.1393$
Generation	$T_{H1_1} + T_{H1_2} + 2T_{AE_1} + T_{Ver_2} + T_{Sig_1} + T_L + T_{S1} + 2T_{AD_1} + T_{AD_2} = 0.5944$
Accountability	$2T_{AE_1} + 2T_{AD_1} + 21T_L + T_{S1} = 0.7017$

Table 4: Computation cost comparison.

Scheme	Generation cost (ms)	Accountability cost (ms)
[17]	$T_P + T_{H2} + T_{S2_1} + T_{S2_2} = 40.5941$	–
[22]	$T_{H1_1} + T_{Sig_1} + T_Z = 18217.9183$	–
[23]	$3T_{AE_2} + 3T_{Sig_1} + 3T_{Ver_1} + 2T_{S1} + 3T_{AD_2} = 0.971$	–
[26]	$T_B + 2T_R = 6.0892$	–
[28]	$T_O + T_{ZKP} = 2350$	$T_M = 1501540$
Ours	$T_{H1_1} + T_{H1_2} + 2T_{AE_1} + T_L + T_{Sig_1} + T_{Ver_2} + 2T_{AD_1} + T_{S1} + T_{AD_2} = 0.5944$	$2T_{AE_1} + 2T_{AD_1} + 21T_L + T_{S1} = 0.7017$

[a]T_P is the PBKDF2 algorithm and the computation cost is 40.18 ms when the parameter size is 8 B and the number of rounds is 61337; T_{H2} indicates the SHA-256 operation, and the computation cost is 0.035 ms when the parameter size is 32 B; T_{S2} denotes the AES-256 encryption/decryption algorithm. T_{S2_1} and T_{S2_2} are the computation cost of performing the AES-256 encryption/decryption algorithm when the parameter sizes are 64 B and 128 B, respectively, and they are 0.187 ms and 0.1921 ms. [b]T_Z denotes the time complexity of zero-knowledge proof. According to the definition of identity information in our scheme, we refer to the performance simulation of [22], and the value of T_Z is 18271.9 ms. [c]T_B serves as the base58 encoding algorithm and the computation cost is 0.086 ms when the parameter size is 256 B; T_R expresses the RSA signature algorithm and the computation cost is 3.002 ms when the parameter size is 450 B. [d]T_O, T_{ZKP}, and T_M indicates the time complexity of an oracle, zero-knowledge proof, and secure multiparty computation, respectively. According to the definition of identity information in our scheme, we refer to the performance simulation of [28], and the value of T_O, T_{ZKP}, and T_M are 1400 ms, 950 ms, and 1501540 ms, respectively.

algorithm are stored locally by the user, which are 304 bytes. And users in [26] need to store a private key for signature algorithm and personal information (name, DID, photo, etc.) locally, which are exceeds 10670 bytes. In [28], a credential containing user's information is stored locally, and the storage cost is estimated to be over 150 bytes. While with the help of the biometrics, there is no data such as keys need to be stored locally by users in our scheme.

For a server, an estimated cost cannot be given in the paper [26, 28], because there is no detailed description. A pair of public and private keys encrypted by the AES-256 algorithm needs to be stored in the server in [17], and the storage cost is 108 bytes. In [22], the user's identity information and related certificates are stored in the server, which are totally 198 bytes. And the server in [23] needs to store the user's information and the certificate containing the user's phone number, photo, and so on, and the storage cost exceeds 10240 bytes. While the storage cost is 505 bytes composed of *IRIs* in our scheme. Because we divide the encrypted user information into multiple pieces based on the shamir (t, n) threshold algorithm and store them in different servers, so as to audit malicious users without revealing user privacy.

The last is the storage cost of each scheme on the blockchain. Since there is no evidence that a blockchain is deployed in [28], an estimated overhead cannot be given. In [17], a public part of the claim used to proof the identity is written into the blockchain and the cost exceeds 200 bytes. In [22], the hash value of the user's identity information and the corresponding certificate are recorded in the blockchain, which is $20 + 82 = 102$ bytes. And in [23], the hash value of the user's identity information is written into the blockchain and the storage cost is 20 bytes. A DID proof with DID, name, signature, etc. Is recorded in the blockchain in [26], which is 800 bytes. In our scheme, the hash value of user's metadata, biometrics, and digital identity, as well as the timestamp are recorded on the blockchain, which is $20 + 20 + 20 + 20 + 14 = 94$ bytes. From Table 5, we achieve lower storage cost in the blockchain compared to the schemes [17, 22, 26]. Although Zheng et al. [23] have a lower overhead than us, the data recorded on the blockchain in our scheme more intuitively shows the entire process of identity generation and accountability. In addition, the data is written to the blockchain by different entities, which decentralizes the power of regulatory authorities and reduces the risk of information leakage compared to [23]. Besides, with the storage requirement of 94 bytes per person, even if the identity information of about seven billion people around the world needs to be stored, the required storage space does not exceed 0.7 TB, which is within the acceptable range for practical use. In short, we liberate users in terms of storage, while servers and the blockchain have the necessary storage requirements to balance privacy and accountability, which are low enough for practical scenarios.

6.4. Blockchain Gas Cost. Considering Gas cost as an important aspect to measure performance, we conduct detailed experiments in this regard. To visualize the execution cost of our smart contract, we evaluate its practical performance on a public Ethereum testnet (Rinkeby). We used the plugin Metamask in Chrome v100.0 explorer to access the Rinkeby testnet and the Remix, a browser-based IDE, to compile and deploy our smart contract. Rinkeby is built in April 2017 by the Ethereum Foundation and it uses the proof-of-authority consensus mechanism. Since the ether supply is controlled by several trusted parties and only they can write

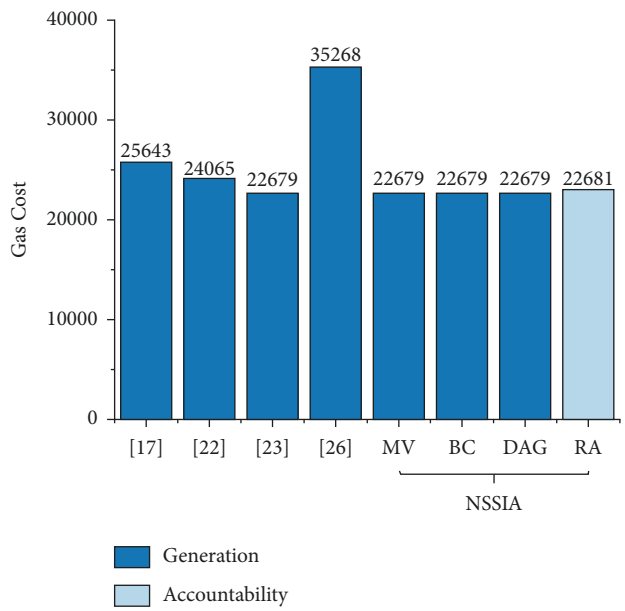


Figure 7: Comparison of gas cost when writing data.

Table 5: Storage cost comparison.

Scheme	User (bytes)	Server (bytes)	Blockchain (bytes)
[17]	168	108	> 200
[22]	16 ~ 32	198	102
[23]	304	> 10240	20
[26]	≈ 10670	—	800
[28]	> 150	—	—
Ours	0	505	94

transactions on the blockchain, it can be considered a consortium blockchain. Hence, the waiting time for a transaction to be confirmed is relatively short to be ignored.

Writing and reading are the main interactions between entities and the consortium blockchain. Therefore, we record the proof data by deploying a smart contract on Rinkeby and count the Gas spent on contract deployment and invocation. And, since Maram et al. [28] have no blockchain deployed, we compare our scheme with the above SSI schemes [17, 22, 23, 26] based on the data in Section 6.3, as shown in Figure 7.

As we can see from Figure 7, during the generation phase, there are more than 200 bytes of data recorded on the blockchain in [17], which costs approximately 25643 Gas. In [22], a total of 102 bytes of data are written to the blockchain and the cost is 24065 Gas. Zheng et al. [23] record 20 bytes of certificates on the blockchain, costing 22679 Gas. The cost of recalling the contract to write 800 bytes of identity proof in [26] is 35268 Gas. In phase 4.1, the contract in our NSSIA is deployed and the cost is 176335 Gas. Unlike the above schemes where only one entity interacts with the blockchain, in our scheme, different entities are responsible for interacting with the blockchain at different stages of identity generation described in Section 4. Concretely, in the digitization stage, metadata verifier (MV) and biometric collector (BC), respectively, record 20-byte metadata and biometric proofs into the blockchain, with an overhead of 22679 Gas. And, in the generation stage, the proof of the generated digital avatar-i (DA) is written into the blockchain by the digital avatar-i generator (DAG), which also costs 22679 Gas. Although the total cost is $22679 * 3 = 68037$ Gas which is greater than other schemes, the Gas cost of our scheme is actually the lowest due to the spread over different entities. In addition, we greatly reduce the risk of centralization of power compared to other schemes.

In terms of accountability, the overhead of the above schemes is 0 due to the lack of accountability mechanism. In our scheme, there are 34 bytes of log information recorded by RA on the blockchain, and the cost is 22981 Gas, which is almost the lowest compared with the overhead of the generation phase. All in all, regardless of the generation stage or the accountability stage, the gas cost of our scheme is not prohibitive for practice use. Furthermore, the decentralization of regulatory authorities' power guarantees fair audits and protects user privacy.

7. Conclusion and Future Work

A new self-sovereign identity scheme with accountability is proposed in this paper, where the executable code is introduced to allow each user to independently control their own identity, referred as the digital avatar-i (DA), and malicious users can be fairly regulated without violating the privacy of legitimate ones. For concreteness, one and only individual-specific executable code is generated for each user to interact with others in metaverse without a third-party program, in which biometrics are integrated into the code to enhance uniqueness and user control. The hash of the individual-specific executable code is used as an identifier and each user can store, read and prove identities with service providers through his/her own local executable code. Furthermore, a joint accountability mechanism is introduced to balance the privacy and accountability, where shamir(t, n) threshold algorithm is used to decentralize the power of each regulatory authority and hide users' information in reality, and the impartial audit is further guaranteed by a consortium blockchain. The security analysis illustrates that our NSSIA can resist multiple security threats such as sybil attacks, impersonation attacks, and so on. And the analysis result on SSI properties shows that we have satisfied all the six SSI properties in the identity generation phase. Compared with the state-of-the-art schemes, the extensive experiment results in performance indicate that the overhead of our NSSIA is not unreasonable for practical use.

For future work, we will pay attention to the difficulties existing in the use of the DA, such as unlinkability, and right to be forgotten, and the full design of Section 4.4 will be presented. Meanwhile, striking a balance between privacy and accountability when using the DA to interact with others in cyberspace is also the focus of our research.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest in this work.

Acknowledgments

This work was supported by the Natural Science Foundation of Zhejiang Province (Grant no. LQ20F020019) and the Foundation of Science and Technology on Communication Security Laboratory (Grant no. 6142103190105).

References

[1] A. Mühle, A. Grüner, T. Meinel, and C. Meinel, "A survey on essential components of a self-sovereign identity," *Computer Science Review*, vol. 30, pp. 80–86, 2018.

[2] Y. Liu, Z. Zheng, G. Guo, W. Xingwei, and T. Zhenhua, "An identity management system based on blockchain," in *Proceedings of the 2017 15th Annual Conference on Privacy, Security and Trust (PST)*, pp. 44–4409, IEEE, Calgary, AB, Canada, 28-30 August 2017.

[3] R. Soltani, U. Trang Nguyen, and A. An, "A survey of self-sovereign identity ecosystem," *Security and Communication Networks*, vol. 2021, 2021.

[4] P. J. Windley, "Sovrin: an identity metasystem for self-sovereign identity," *Frontiers in Blockchain*, vol. 4, no. 30, 2021.

[5] P. Dunphy and F. A. P. Petitcolas, "A first look at identity management schemes on the blockchain," *IEEE security & privacy*, vol. 16, no. 4, pp. 20–29, 2018.

[6] A. J. Zwitter, O. J. Gstrein, and E. Yap, "Digital identity and the blockchain: universal identity management and the concept of the "Self-Sovereign" individual," *Frontiers in Blockchain*, vol. 3, no. 26, 2020.

[7] C. Allen, *The Path to Self-Sovereign Identity*", http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereign-identity.html, 2016.

[8] S. Manski, "Distributed ledger technologies, value accounting, and the self sovereign identity," *Frontiers in Blockchain*, vol. 3, no. 29, 2020.

[9] S. S. Darnell, J. Sevilla, and S. Joseph, "3 stages of a pan-african identity framework for establishing self-sovereign identity with blockchain," *Frontiers in Blockchain*, vol. 4, no. 26, 2021.

[10] A. Grech, I. Ariño, and L. Ariño, "Blockchain, self-sovereign identity and digital credentials: promise versus praxis in education," *Frontiers in Blockchain*, vol. 4, no. 7, 2021.

[11] A. Boysen, "Decentralized, self-sovereign, consortium: the future of digital identity in Canada," *Frontiers in Blockchain*, vol. 4, no. 11, 2021.

[12] C. T. Kalman and A. Anderson-Priddy, "Self-sovereign digital identity: a paradigm shift for identity," *IEEE Security & Privacy*, vol. 17, no. 3, pp. 17–27, 2019.

[13] G. Kondova and J. . Erbguth, "Self-sovereign identity on public blockchains and the gdpr," in *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, pp. 342–345, March 30- April 3 2020.

[14] F. Schardong and R. Custódio, *Self-sovereign Identity: A Systematic Map and Review*, Available: https://arxiv.org/abs/2108.08338, 2021.

[15] F. De Filippi and P. De Filippi, "Self-sovereign identity in a globalized world: credentials-based identity systems as a driver for economic inclusion," *Frontiers in Blockchain*, vol. 2, no. 28, 2020.

[16] M. Freytsis, I. Barclay, S. K. Radha et al., "Development of a mobile, self-sovereign identity approach for facility birth registration in Kenya," *Frontiers in Blockchain*, vol. 4, no. 2, 2021.

[17] M. Takemiya and B. Vanieiev, "Sora identity: secure, digital identity on the blockchain,"vol. 2, pp. 582–587, in *Proceedings of the 2018 IEEE 42nd annual computer software and applications conference (compsac)*, vol. 2, pp. 582–587, IEEE, Tokyo, Japan, 23-27 July 2018.

[18] T. Zhou, X. Zhao, and H. Zhao, "Everssdi: blockchain-based framework for verification, authorisation and recovery of self-sovereign identity using smart contracts," *International Journal of Computer Applications in Technology*, vol. 60, no. 3, pp. 281–295, 2019.

[19] J. Niu and Z. Ren, "A self-sovereign identity management scheme using smart contracts," in *Proceedings of the MATEC Web of Conferences*, vol. 336, EDP Sciences, Article ID 08005, 15 February 2021.

[20] M. Westerkamp, S. Göndör, and A. Küpper, "Tawki: towards self-sovereign social communication," in *Proceedings of the 2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON)*, vol. 29–38, IEEE, Newark, CA, USA, 04-09 April 2019.

[21] Q. Stokkink and J. Pouwelse, "Deployment of a blockchain-based self-sovereign identity," in *Proceedings of the 2018 IEEE international conference on Internet of Things (iThings) and IEEE green computing and communications (GreenCom) and IEEE cyber, physical and social computing (CPSCom) and IEEE smart data (SmartData)*, pp. 1336–1342, IEEE, Halifax, NS, Canada, 30 July 2018 - 03 August 2018.

[22] J. Lee, J. Hwang, J. Choi, H. Oh, and J. K. Sims, "Self sovereign identity management system with preserving privacy in blockchain," *IACR Cryptol. ePrint Arch.*vol. 1241, 2019 Available: https://eprint.iacr.org/2019/1241.

[23] Y. Zheng, Y. Li, Z. Wang, C. Deng, and Y. Luo, "Blockchain-based privacy protection unified identity authentication," in *Proceedings of the 2019 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, pp. 42–49, IEEE, Guilin, China, 17-19 October 2019.

[24] T. Hamer, K. Taylor, K. S. Ng, and A. Tiu, *Private Digital Identity on Blockchain*BlockSW/CKG@ ISWC, 2019.

[25] A. Othman and J. Callahan, "The horcrux protocol: a method for decentralized biometric-based self-sovereign identity," in *Proceedings of the 2018 international joint conference on neural networks (IJCNN)*, pp. 1–7, IEEE, Rio de Janeiro, Brazil, 08-13 July 2018.

[26] E. Bandara, X. Liang, F. Peter, S. Shetty, and K. De Zoysa, "A blockchain and self-sovereign identity empowered digital identity platform," in *Proceedings of the 2021 International Conference on Computer Communications and Networks (ICCCN)*, pp. 1–7, IEEE, Athens, Greece, 19-22 July 2021.

[27] Q. Stokkink, G. Ishmaev, D. Epema, and J. Pouwelse, "A truly self-sovereign identity system," in *Proceedings of the 2021 IEEE 46th Conference on Local Computer Networks (LCN)*, pp. 1–8, IEEE, 2021.

[28] D. Maram, H. Malvai, F. Zhang, N. Jaean-Louis, and A. Frolov, "Candid: can-do decentralized identity with legacy compatibility, sybil-resistance, and accountability," in *Proceedings of the 2021 IEEE Symposium on Security and Privacy (SP)*, pp. 1348–1366, IEEE, San Francisco, CA, USA, 24-27 May 2021.

[29] Y. Wang, S Zhou, and N. Zhang, *A Survey on Metaverse: Fundamentals, Security, and Privacy*, Available: https://arxiv.org/abs/2203.02662, 2022.

[30] L.-H. Lee, T. Braud, and P. Zhou, "All one needs to know about metaverse: a complete survey on technological singularity, virtual ecosystem, and research agenda," Available: https://arxiv.org/abs/2110.05352, 2021.

[31] Q. Lyu, S. Cheng, and H. Li, *Nssia: A New Self-Sovereign Identity Scheme with Accountability*, Available: https://arxiv.org/abs/2206.04911, 2022.

[32] M. Sporny, D. Longley, and M. Sabadello, *Decentralized Identifiers (DIDs) v1.0*, https://www.w3.org/TR/did-core/, 2021.

[33] N. Otto, S. Lee, and B. Sletten, *Verifiable Credentials Use Cases*, https://www.w3.org/TR/vc-use-cases/, 2019.

[34] D. Yao and A. Yao, "On the security of public key protocols," *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, 1983.

[35] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612-613, 1979.

[36] Q. Lyu, Y. Qi, X. Zhang, H. Liu, Q. Wang, and N. Zheng, "Sbac: a secure blockchain-based access control framework for information-centric networking," *Journal of Network and Computer Applications*, vol. 149, Article ID 102444, 2020.

[37] A. Sudan and M. Sudan, "A fuzzy vault scheme," *Designs, Codes and Cryptography*, vol. 38, no. 2, pp. 237–257, 2006.

[38] Md S. Ferdous, F. Alassafi, and M. O. Alassafi, "In search of self-sovereign identity leveraging blockchain technology," *IEEE Access*, vol. 7, pp. 103059–103079, 2019.

[39] K. Bobkowska, K. Przyborski, and M. Przyborski, "Incorporating iris, fingerprint and face biometric for fraud prevention in e-passports using fuzzy vault," *IET Image Processing*, vol. 13, no. 13, pp. 2516–2528, 2019.