Blockchain Based IIoT Data Sharing Framework for SDN-Enabled Pervasive Edge Computing

Ying Gao, *Member, IEEE*, Yijian Chen, Xiping Hu, *Member, IEEE*, Hongliang Lin, Yangliang Liu, and Laisen Nie

***Abstract*—Pervasive edge computing (PEC) is an emerging paradigm for the industrial Internet of Things (IIoT), and software-defined networks (SDN) offer lower latency services, and massive intelligent devices connectivity for the IIoT. However, the PEC has some issues with data security, and privacy while PEC devices sharing data among edges. What's more, the centralized SDN suffers from single point of attacks such as distributed denial of service (DDoS) from IIoT devices, and has the challenge of data leakage. In this article, we use blockchain, and proxy reencryption (PRE) technologies to tackle these challenges. The blockchain authorizes all devices in the network to improve their credibility, and authenticity. In addition, a blockchain-based data sharing framework that combines a PRE scheme is introduced for secure device-to-device communication in PEC environments. A series of smart contracts are designed for flexible operations of searching, and updating records on the blockchain. The experiments reveal that our design is highly efficient, and has high performance.**

***Index Terms*—Blockchain, data sharing, pervasive edge computing (PEC), proxy reencryption (PRE), software-defined networks (SDN).**

I. Introduction

INDUSTRIAL Internet of Things (IIoT) is a significant application in Industrial 4.0, including machines, computers, and people enabling intelligent industrial operations [1]. Pervasive edge computing(PEC) is a kind of edge computing techniques that allows Internet of Things (IoT) devices generating, sharing, and processing data among peer edge servers, such as embedded devices, mobile phones, and automatical machines locally [2]. Nevertheless, the PEC technique has some security and privacy concerns and limited performance on networking.

Manuscript received May 15, 2020; revised July 10, 2020; accepted July 13, 2020. Date of publication July 28, 2020; date of current version April 2, 2021. This work was supported in part by the Key R&D Program in Key Areas of Guangdong Province under Grant 2019B010137001 and Grant 2020B010166001, in part by the Industrial Internet Innovation and Development Project in 2018 under Grant MIZ1824020, and in part by the Guangzhou Key Industrial Technology Project under Grant 201802020014. Paper no. TII-20-2480. *(Corresponding author: Yijian Chen; Xiping Hu.)*

Ying Gao, Yijian Chen, Hongliang Lin, and Yangliang Liu are with the School of Computer Science and Engineering, South China University of Technology, Guangzhou 510641, China (e-mail: gaoying@scut.edu.cn; csyjchan@foxmail.com; cslhlin@mail.scut.edu.cn; 201821034188@mail.scut.edu.cn).

Xiping Hu is with the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Beijing 100864, China (e-mail: xp.hu@siat.ac.cn).

Laisen Nie is with the School of Electronics and Information, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: nielaisen@nwpu.edu.cn).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2020.3012508

In the IIoT environment, the PEC architecture consists of three layers generally, including cloud layer, pervasive edge layer, and IIoT device layer. There are several challenges in the PEC. First, the cloud servers may be curious or malicious to extract sensitive information of users due to their control of users' data [3]. Second, data in the edge servers have a higher probability of being tampered with. This is because the edge has less data security measures compared with the central cloud servers [3]. Third, data are sharing and transferring among the edge and IIoT devices frequently, which needs sufficient protection for data. A traditional solution to guarantee data security and privacy is using cryptographies. Nonetheless, little work has been devoted to investigating how to adopt crypto methods to assure data security in PEC.

Besides, to enhance the performance on networking in PEC, software-defined networking(SDN) is considered as enabling technologies for PEC [4]. The SDN decouples the data plane and the control plane through an SDN controller for users [5]. Nonetheless, the SDN controller is centralized and suffers from a potential single point of attacks such as distributed denial of service (DDoS) [6]. Attackers can control the compromised and unauthorized IIoT devices to intrude the SDN controller and lead to data leakage. A traditional solution is to build an authentication system to authenticate devices. However, the system is centralized and has several issues like single point of failure, data being tampered with, data leakage, etc.

To address the issues mentioned previously, we combine blockchain technology [7] and a cryptographic primitive called proxy reencryption (PRE) [8] in our design.

Blockchain technology is claimed to be a trustless distributed ledger in both academia and industry [9][10]. Records on the blockchain are traceable and cannot be tampered with due to blockchain's unique block structure and special consensus mechanisms. There are three kinds of blockchain typically, public blockchain, consortium blockchain, and private blockchain. The consortium blockchain is used in associations that consist of several companies or organizations and there is a certificate authority (CA) to verify the participants' identities and grant access control for them. We adopt a consortium blockchain to authenticate IIoT devices in PEC environments. The blockchain

CA acts as the traditional authentication system and solves the problems of the system mentioned previously.

PRE is a public key cryptographic primitive that a data owner can delegate the ability to decrypt his data to other data requestors [11]. After a semi-trusted proxy reencrypts the ciphertext under an owner's public key, a data requestor is able to decrypt the new ciphertext through his secret key. Identity-based proxy reencryption (IBPRE) [12] is a kind of PRE scheme that data owners and requestors take their identities as public keys and do not need public key infrastructure (PKI) anymore.

We adopt the IBPRE algorithm in this article for three reasons. First, the IBPRE scheme effectively ensures the security and privacy of data sharing between two entities in the heterogeneous and untrusted network. Therefore, the scheme is suitable for data sharing between devices in the PEC-enabled IIoT environment. Second, devices encrypt their data by IBPRE and store them in the cloud servers. They do not need to download the encrypted data while sharing with other devices, which simplifies the process of sharing data compared with other schemes [13]. Third, the IBPRE is an improvement of PRE, which gets rid of PKI and simplifies the certificate management in PRE [12].

The IBPRE, however, requires a private key generator (PKG) to authenticate users, generate private keys for users and maintain the keys. If curious or malicious members control the PKG, they will get the ability to access users' data. To address this issue, the blockchain is used to manage encryption keys for device data and record the access of keys to realize accountability and enhance data security.

Our contributions are summarized as follows.

1) We design a secure and privacy-preserved data sharing framework that combines the IBPRE scheme with a consortium blockchain under the SDN-enabled PEC environment. What's more, a comprehensive case study is involved to demonstrate the whole data sharing process.

2) The blockchain technology is used to authorize IIoT devices and provide reliable and credible IIoT devices connectivity for the SDN-enabled PEC environment to improve IIoT network security. Besides, the PKG in IBPRE is also monitored by the blockchain.

3) We design several smart contracts for querying and updating crypto keys for IBPRE on the blockchain. The experiments and security analysis indicate that our system has well performance and security.

The rest of this article is organized as follows. Section II introduces some related work about our research. Section III is the demonstrated system model in detail. In Section IV, we do a case study about our proposed model to verify its feasibility. In Section V, we simulate the model and do some experiments to analyze the performance. In Section VI, we analyze the security of our design. Finally, Section VII concludes this article.

II. Related Work

A. Blockchain for PEC and SDN

Less study focused on using blockchain in PEC technology. Huang *et al.* [2] designed a blockchain system in PEC environments to reduce storage and energy consumption by their proposed new by their proposed new Proof of Stake(PoS) mechanism. The blockchain technology was used to guarantee trust between peer nodes on sharing micro-payment and microaccess control information in PEC. However, this study did not focus on the data security and privacy, which may lead to some concerns.

Most research worked on blockchain and edge computing technology [14]–[17]. The article [17] adopted blockchain and edge computing techniques to build a distributed and trusted authentication framework. The framework achieves trusted authentication and activity traceability of edge servers. BlockTDM [18] is a blockchain-based trust data management scheme with mutual authentication protocol. It aims to enhance data security and privacy for edge terminals or IIoT devices.

Sharma *et al.* [19] proposed a cloud framework for IoT using blockchain, SDN, and fog computing technologies. The blockchain was adopted to record payments of cloud resource transactions between service providers and users. The SDN enabled with the fog nodes were set up in the edge network for IoT devices' data processing locally. Since this framework was implemented in the public blockchain, it has the challenges of authentication and certification management.

Trustlist [20] combines SDN and blockchain to enforce IoT traffic management automatically at the edge networks. Hence, the design can improve the trustability and authenticity of IoT devices. Moreover, the scalability and security of the whole IoT network can be enhanced.

B. Proxy Reencryption

PRE was first proposed about two decades ago [8]. The primitive become popular in providing data confidentiality and access control in cloud computing.

SDSM is a secure data service mechanism in mobile cloud computing [21]. The mechanism utilizes the IBPRE and allowed the cloud service providers to manage data delegation for users. This means the the cloud service provider (CSP) is able to manage the access control to data and users have to trust that the cloud is not curious about their data.

Nucypher [22] is an active project in the community, which is declared as a decentralized key management system(KMS). It aims to solve the issue of trusted and centralized KMS in the cloud. It manages encryption keys and share data between two entities based on blockchain.

III. System Framework

A. Overview

Fig. 1 demonstrates our blockchain-based data sharing framework in the SDN-enabled PEC for IIoT. It consists of four parts, the IIoT environment, the SDN layer, the PEC environment, and the blockchain network.

The bottom of the framework is the IIoT environment, including smart factories, smart manufactories, automatical control systems, industrial sensors, etc. They generate producing data, processing data, and monitoring data continuously and deliver them to the edge servers.

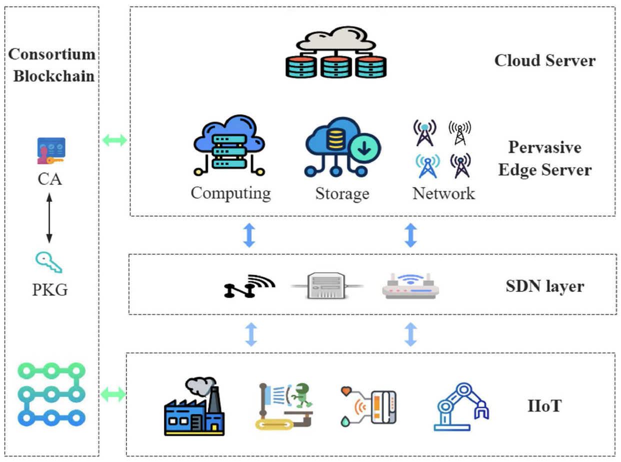

Fig. 1. System framework.

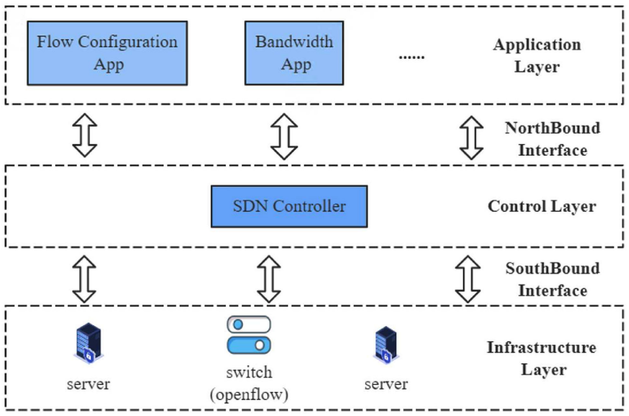

Fig. 2. Architecture of SDN.

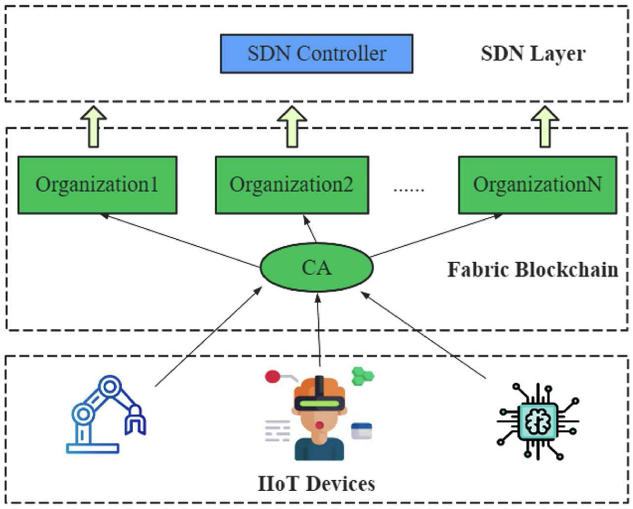

Fig. 3. Device authorization in Hyperledger Fabric for SDN.

The SDN layer abovementioned the IIoT environment is enabled with the SDN controller and offers programmable network with some protocols such as Openflow [23]. Data transmitted through this layer will improve the network performance.

Upside the SDN layer, there is the PEC environment which consists of the cloud servers and edge servers. Cloud servers include some central servers from cloud service providers and network operators, which controls the interaction of the entire edge computing service. In addition, the edge server provides IIoT devices with computing and storage resources locally, which can reduce the latency of network operations and service interactions.

The blockchain network integrates the IBPRE to assure the security and privacy of devices' data. It manages the encryption keys of the IBPRE securely. What's more, the blockchain records the entire process of sharing data between two devices. Data on the blockchain are transparent, auditable, and traceable.

B. Blockchain Empowered SDN in PEC Environments

In our proposed framework, we adopted Hyperledger Fabric,[1,2] a kind of consortium blockchain, as the blockchain network. The fabric CA in Fabric generates certificates and private keys for members.

Fig. 2 is the architecture of an SDN network, including infrastructure layer, control layer, and application layer. The infrastructure layer consists of various network devices, such as a set of network switches, and routers. The control layer controls the physical devices in the infrastructure layer through the southbound interface which is enabled with protocols like Openflow. The application layer includes a series of applications for network functions, such as flow configuration, bandwidth configuration, etc.

Fig. 3 is the process of IIoT device registration and authorization in fabric CA to improve the security of SDN. In the PEC environments, the SDN handles massive IIoT devices and offers flow forwarding service. All IIoT devices have to register in the fabric CA to get their own certificates and keys to enhance reliability when they first come into the system. This will overcome the issue of the issue of DoS(Denial of Service)/DDoS attacks to the SDN network.

[1][Online]. Available: https://www.hyperledger.org/
[2][Online]. Available: https://cn.hyperledger.org/projects/fabric

C. Blockchain and IBPRE Empowered Secure Data Sharing for PEC

1) PRE and IBPRE: Assume device A as Alice, which is the data owner, and device B as Bob, the data requestor. A general process of PRE between two mobile devices is shown in the Fig. 4.

1) Device A encrypts its data M with its public key pk_A, which is its identity, and gets ciphertext C_A.
2) When device A wants to share M with device B, it generates a reencryption key $Rekey$ by its private key sk_A and B's public key pk_B, which is also its identity.
3) A sends C_A and Rekey to the proxy, which can be an edge server.
4) The proxy reencrypts the C_A with Rekey to get C_B and sends C_B to device B.

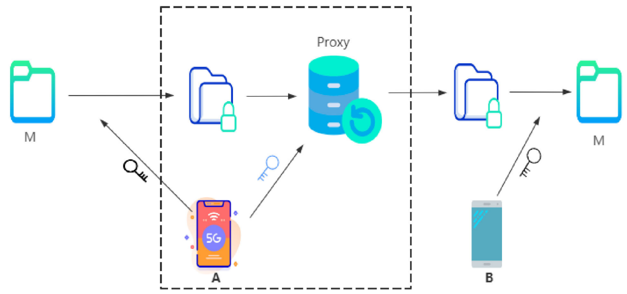

Fig. 4. General process of PRE.

5) B decrypts C_B with its private key sk_B and finally obtains the data M.

In this process, A just encrypts the data once to get the cipher. whenever someone requests A's data, A just need to send the cipher and the reencryption key to the proxy to do the reencryption step, which is shown in the dotted frame. Therefore, A does not need to download and decrypt the cipher. This simplifies the process of sharing compared with traditional way.

The identities can be device ID or serial number from the device manufacturer. Besides, the identities can derive from blockchain when the devices are registered in the blockchain. In the IBPRE, a PKG is required to authenticate users, generate private keys for users and maintain the keys.

We combine the IBPRE algorithm with the Fabric platform in this framework. First, the IBPRE is adopted to share data securely between two entities in a complicated network. The Fabric is used to store crypto keys of IBPRE and record all events happened during the sharing process. Second, we manage the PKG in IBPRE through Fabric CA. In our system, the PKG is a service controlled by the CA, which only generates keys for users. Third, Fabric CA is used to authenticate users, send requests to PKG, and receive the responses from PKG.

2) Blockchain and IBPRE for Secure Data Sharing Scheme: There are four entities in our data sharing framework, including *Data Owner*, *Data Requestor*, *Proxy Server*, and *Hyperledger Blockchain:*

Data Owner: users who hold the data to store or share. In the PEC scenario, they can be pervasive edge servers which collect and store data from IIoT network.

Data Requestor: users who requesting for data owner's data. They can be other edge servers from homogeneous or heterogeneous domains in the PEC environments.

Data Server: the data server can be a cloud server or an edge server. It stores encrypted data from data owners and acts as a proxy to run the reencryption step in the IBPRE.

Hyperledger Blockchain: The blockchain contains two functions in this model. First, storing encryption keys and other vital data to achieve security. Second, maintaining records in the sharing process for auditing and provenance.

The system framework is shown simply in Fig. 5. Table I illustrates some notations adopted throughout the article.

We involve seven phases in this system: *Setup*, *UserRegistration*, *Encrypt*, *ReKeyGen*, *ReEncrypt*, *FirstDecrypt*, *SecondDecrypt*: The descriptions of different phases are summarized as follows:

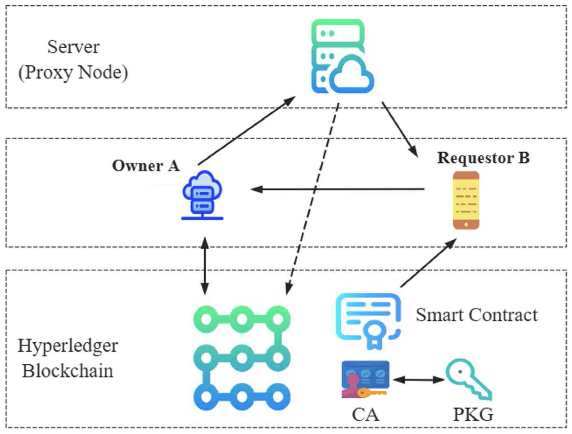

Fig. 5. Blockchain and IBPRE framework.

TABLE I
NOTATIONS

Notation	Description
ID_A, Sk_A	Data owner's public key and private key
ID_B, Sk_B	Data requestor's public key and private key
F	Data for storing or sharing
F_k	Symmetric key for data encryption
F_{hash}	Data hash
F_{kw}	Data keywords
CT_f	Encrypted Data
CT_{fk}	Encrypted symmetric key
$Rekey_{A\to B}$	Re-encryption key from owner to a requestor
CT_{rfk}	Re-encrypted CT_{fk}

1) Setup: In this phase, the system is initialized and produces public parameters for the IBPRE. The Fabric network is launched and connects to the PKG service with a master key.
2) UserRegistration(ID): All IIoT devices and edge servers register in the Fabric CA with specific IDs. They will obtain certificates and keys for Fabric and private keys generated by PKG for IBPRE.
3) Encrypt $(F, F_k, ID_A) \to (CT_f, F_{kw}, CT_{fk})$: This phase includes three steps.
   a) FileEncrypt $(F, F_k) \to (CT_f)$: Owner A encrypts its data F with a symmetric key F_k to get the ciphertext CT_f. It then uploads the CT_f to the data servers.
   b) KeyEncrypt $(F_k, ID_A) \to (CT_{fk})$: Owner A encrypts F_k with its public key ID_A to obtain the ciphertext of F_k, denoted as CT_{fk}.
   c) InfoUpload $(F_{kw}, F_{\text{hash}}, CT_{fk})$: Owner A computes the data F's hash F_{hash}, chooses some keywords F_{kw} related to the data and stores F_{hash}, CT_{fk}, and F_{kw} into Fabric.
4) ReKeyGen $(Sk_A, ID_B) \to \text{Rekey}_{A\to B}$: After receiving a request from requestor B, A first checks whether there is a reencryption key $\text{Rekey}_{A\to B}$ in Fabric. If exists, it receives the key from Fabric. Otherwise, A generates a new reencryption key with its private key Sk_A and B's public key ID_B. Finally it uploads the new key to Fabric.
5) ReEncrypt $(CT_{fk}, \text{Rekey}_{A\to B}, F_{kw}) \to CT_{rfk}$:

a) Owner A sends $\text{Rekey}_{A\to B}$ and the encrypted symmetric key CT_{fk} to the data server which acts as a proxy.

b) The server reencrypts CT_{fk} with $\text{Rekey}_{A\to B}$ to get CT_{rfk} and records it with F_{kw} into Fabric. It then sends F_{kw} to B.

6) FirstDecrypt $(F_{kw}, Sk_B) \to (F_k, F_{\text{hash}})$*:* In this phase, requestor B queries the blockchain with F_{kw} and obtains CT_{rfk}. The CT_{rfk} will be decrypted into F_k and F_{hash} by B's private key Sk_B.

7) SecondDecrypt $(F_k, F_{\text{hash}}) \to F$*:* Requestor B downloads the encrypted data CT_f from the data server and decrypts it with F_k. Next, B computes the file hash again and checks whether the new file hash is equal to F_{hash}. If equal, B finally obtains the file it requests for.

3) Smart Contract Design for Secure Data Sharing in the SDN-Enabled Smart Communities: We design four smart contracts for the system, including $queryKeyByKw$, $queryRekey$, $queryRCTByKw$, $updateKw$. They are introduced as follows.

$queryKeyByKw(ownerId, keywords)$*:* This function will return specific encrypted file encryption key CT_{fk} and the file hash F_{hash} for ownerId. Notice that ownerId must map to the contract caller. The function then searches in Fabric to collect all corresponding records associated with the keywords.

$queryRekey(ownerId, requestorId)$*:* This function will return a key for reencryption from data owner to the data requestor. Considering to secure access to the reencryption key, the algorithm is only supposed to be executed by ownerId. The contract will then query a specific reencryption key on the blockchain.

$queryRCTByKw(requestorId, keywords)$*:* This function will return specific reencrypted file encryption key CT_{rfk} and the data hash F_{hash} for requestorId. The contract is also only called by requestorId who will be authenticated by Fabric CA at the beginning.

$updateKw(id, keywords, newKeywords)$*:* This function replaces the old keywords with the specific new keywords. The contract first reads records with keywords as a key on the blockchain, and then store the records back to the blockchain with the new Keywords as a new key. However, once the function is executed maliciously, such as replacing the records with irrelevant keywords, users cannot query the records with the old keywords anymore. To avoid malicious tampering, user authentication will be performed by CA.

IV. Case Study

In this section, we will demonstrate our framework in detail in the PEC data sharing scenario. Noticed that the adopted IBPRE scheme is based on Green and Atenieses' first attempt [12]. Since the scheme is multiple use, the data requestor can reencrypt the ciphertext again for a third requestor, which might cause some concerns in the specific IIoT data sharing scenario. Therefore, we constrain the scheme to be single-use in our implementation for more suitability.

Suppose that a data owner named Alice wishes to share her IIoT data with a requestor Bob securely. Denote a user's identity (ID) as the form Institution||Role||Name. For instance, the ID of Alice who is in Factory A can be

$$ID_A : \text{FactoryA}||\text{OwnerA}||\text{Alice} \tag{1}$$

and the ID of Bob in Factory B can be

$$ID_B : \text{FactoryB}||\text{RequestorB}||\text{Bob} \tag{2}$$

Additionally, their private keys are denoted as Sk_A, Sk_B. The complete construction is discussed as follows.

1) Setup$(1^\lambda) \to (PP, MK)$: Let $\mathbb{G}_1$ and $\mathbb{G}_T$ be two cyclic groups of big prime order p. Denote p: $\mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$ as a bilinear map and g as a generator of $\mathbb{G}_1$. Construct two collision-resistant hash functions $\mathcal{H}_0$: $\{0,1\}^* \to \mathbb{G}_1$ and $\mathcal{H}_1 : \mathbb{G}_T \to \mathbb{G}_1$. Output system public parameters.

$$PP = \{\mathbb{G}_1, \mathcal{H}_0, \mathcal{H}_1, g, g^s\} \tag{3}$$

Select a random $s \in \mathbb{Z}_q^*$ as the system master key MK and output $MK = \{s\}$. The PP is broadcast in the system and the MK is kept by the PKG secretly. Blockchain network is then initialized and some smart contracts are deployed on it for searching records with keywords. The PKG is connected to the Fabric CA.

2) UserRegistration(PP, ID): Alice and Bob both register in the Fabric and receive their private keys and certificates for Fabric. Whenever users access to the blockchain for recording or querying, the Fabric CA will verify their certificates. In addition, Alice and Bob obtain private keys for the IBPRE by sending their IDs to the CA. The PKG computes

$$Sk_A = \mathcal{H}_0(ID_A)^s \tag{4}$$

$$Sk_B = \mathcal{H}_0(ID_B)^s \tag{5}$$

and returns Sk_A and Sk_B to the users through CA.

3) Encrypt $(PP, F, F_k) \to (CT_f, F_{kw}, CT_{fk}, F_{\text{hash}})$:

a) FileEncrypt $(F, F_k) \to (CT_f, F_{\text{location}})$: Alice first computes the file hash.

$$F_{\text{hash}} = H_{512}(F) \tag{6}$$

where H_{512} denotes a collision-safe secure hash algorithm $SHA-512$. Next, she generates a symmetric key F_k by advanced encryption standard (AES) algorithm and then encrypts her IIoT data F with F_k and obtains the ciphertext CT_f. Last, she uploads CT_f to the cloud storage and keeps the file's location F_{location}.

b) KeyEncrypt $(PP, F_k, F_{\text{location}}, ID_A) \to (CT_{fk})$: Denote F_k and F_{location} as message M, select a random $r \in \mathbb{Z}_q^*$. Alice computes

$$C_1 = g^r \tag{7}$$

$$C_2 = M \cdot e(g^s, \mathcal{H}_0(ID_A))^r \tag{8}$$

and outputs

$$CT_{fk} = C_1||C_2 \tag{9}$$

The CT_{fk} is the ciphertext of the file encryption key F_k, which can only be decrypted by Alice by computing $C_2/e(C_1, Sk_A)$.

c) InfoUpload $(F_{kw}, F_{\text{hash}}, CT_{fk})$: Alice chooses some keywords F_{kw} about her IIoT data. Assump $F_{kw} = (Alice, headache, Augustin2019)$. Alice stores F_{kw}, CT_{fk}, and F_{hash} into Fabric in a specific form through a smart contract. Thus, Alice can query the cipher by inputting one or more keywords.

4) ReKeyGen $(Sk_A,\ ID_B) \rightarrow$ Rekey$_{A\rightarrow B}$: Bob sends a request to Alice for her IIoT data with his ID ID_B. Alice queries the blockchain with ID_B through a smart contract to check whether there is a reencryption key Rekey$_{A\rightarrow B}$ for Bob. If exists, she receives the key from Fabric. Otherwise, Alice will generate a new one by the following procedures. Select a random element in $\mathbb{G}_T$ as X and a random $r' \in \mathbb{Z}_q^*$, compute

$$Rk_1 = g^{r'} \tag{10}$$

$$Rk_2 = X \cdot e(g^s, \mathcal{H}_0(ID_B))^{r'} \tag{11}$$

$$Rk_3 = Sk_A^{-1} \cdot \mathcal{H}_1(X) \tag{12}$$

denote

$$\text{Rekey}_{A\rightarrow B} = Rk_1||Rk_2||Rk_3 \tag{13}$$

Alice then stores Rekey$_{A\rightarrow B}$ into Fabric in a specific form through a smart contract.

5) ReEncrypt $(CT_{fk}, \text{Rekey}_{A\rightarrow B}, F_{kw}) \rightarrow CT_{rfk}$

a) Alice sends Rekey$_{A\rightarrow B}$ and CT_{fk} to the proxy.

b) The proxy runs the reencryption algorithm by doing some computation as follows. The form of CT_{fk} is denoted as $CT_{fk} = C_1||C_2$, and Rekey$_{A\rightarrow B} = Rk_1||Rk_2||Rk_3$. Let

$$RC_1' = C_1 \tag{14}$$

$$RC_3' = Rk_1 \tag{15}$$

$$RC_4' = Rk_2 \tag{16}$$

and compute

$$RC_2' = C_2 \cdot e(C_1, Rk_3) \tag{17}$$

Denote

$$CT_{rfk} = RC_1'||RC_2'||RC_3'||RC4' \tag{18}$$

The proxy then responses to Alice with CT_{rfk}.

c) Alice queries the Fabric with F_{kw} to obtain F_{hash}. She then stores CT_{rfk} with F_{kw} and F_{hash} into the blockchain in a specific form through a smart contract.

6) FirstDecrypt $(F_{kw}, Sk_B) \rightarrow (F_k, F_{\text{location}}, F_{\text{hash}})$: Alice replies Bob with the keywords F_{kw} related to the IIoT data. Bob queries the blockchain with F_{kw} and gets F_{hash} and the reencrypted ciphertext $CT_{rfk} = RC_1'||RC_2'||RC_3'||RC4'$ through the smart contract. Bob computes

$$M_1 = RC_4'/e(RC_3', Sk_B) \tag{19}$$

and then computes

$$M = RC_2'/e(RC_1', \mathcal{H}_1(M_1)) \tag{20}$$

M is the plaintext which Bob requires for, including F_k and F_{location}.

7) SecondDecrypt $(F_k, F_{\text{hash}}, F_{\text{location}}) \rightarrow F$: Bob sends a request for the encrypted IIoT data CT_f to the cloud in location F_{location} and downloads it. He then decrypts CT_f with F_k like $F = Dec_{Fk}(CT_f)$ using AES algorithm. Next, Bob computes $F'_{\text{hash}} = H_{512}(F)$ and checks whether $F'_{\text{hash}} = F_{\text{hash}}$. If equal, Bob finally obtains the IIoT data he requires for.

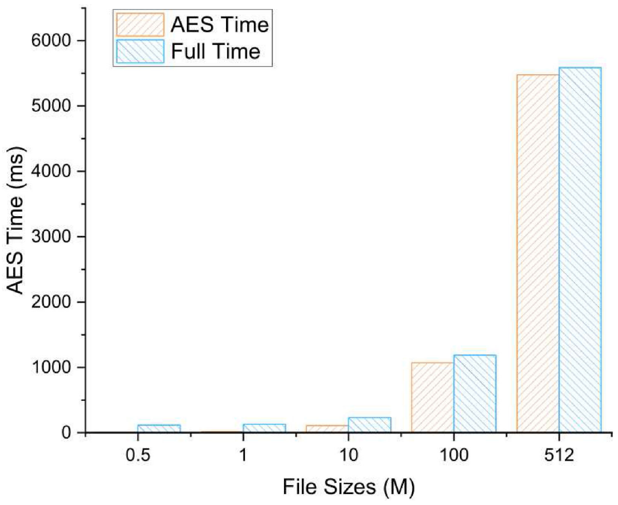

Fig. 6. Time of encryption process in different file sizes.

V. Experiments

In this section, we do a series of experiments about the IBPRE scheme and our designed smart contracts in the blockchain.

Our system consists of two environments, including a cloud server for storage and system control and the SDN network platform. The cloud server environment is Intel Core i7-8700 CPU, 8 GB RAM. We use Mininet for the SDN network simulation.

In the blockchain network, we adopt Hyperledger Fabric in version 1.2.0 and install several nodes, including 1 orderer, 4 peers, and 2 CA for two organizations in the network. A Fabric SDK in Java language is chosen for device registration and enrollment with Fabric CA and calling smart contracts. The IBPRE scheme is utilized by using java paring-based cryptography library. The private key for a specific ID and the AES key for encrypting data are both set to 128 B.

1) Performance of Encryption: To evaluate the performance of file encryption, we design this experiment to test the time cost of encrypting files in various sizes and uploading the crypto keys to the blockchain. The file sizes are set from 512 k to 512 M while the symmetric key size is set to 128 B. Fig. 6 depicts the time cost in AES encryption and the full process time cost including recording keys to Fabric. The result indicates that the offline AES encryption costs most of the full process time, and the cost of interacting with blockchain, which is online, keeps steadily in about 110 ms, which can reduce the network burden.

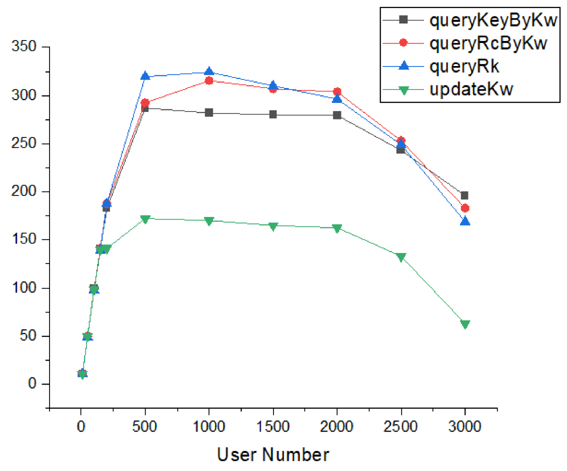

Fig. 7. Throughput of different smart contracts in different concurrent users.

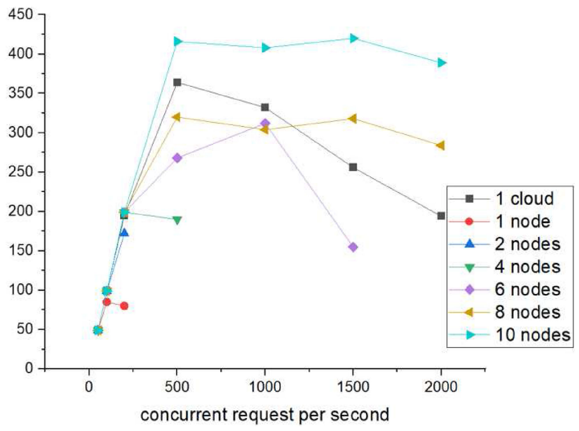

Fig. 8. Throughput of reencryption in cloud and edge environments with different edge nodes.

2) Throughput of Smart Contract Calling: We design four smart contracts for specific operations on the blockchain. Since the smart contracts can be executed by all authenticated members in Fabric, an experiment of contracts executing performance is involved and demonstrates the result in Fig. 7. The throughput is evaluated by transactions per second (TPS), the number of transactions executed in the blockchain in one second. The chart reveals the TPS of calling different contracts in various concurrent user devices. With the increasing number of concurrent device requests, the throughput rises rapidly at first. But when the number reaches to about 500, the TPS of three query contracts seems to reach a saturation point at about 300, and the update contract reaches to approximately 160. When the number achieves to about 2000, the TPS of four contracts begin to fall down because of the limited computation and storage of the experiment environment.

3) Throughput of Reencryption in Cloud and Edge Environment: To evaluate how PEC technology improves system performance, We compare the performance of a cloud center and 2–10 edge nodes in the reencryption stage of IBPRE, and set different concurrent requests from 0 to 2000. The edge node has 1 core CPU and 1 GB RAM while the cloud has 8 core CPU and 8 GB RAM. The results are shown in the Fig. 8. The black line represents the situation of a single cloud center. The TPS increases first and then decreases. This is due to the limited resources in the cloud. The other lines represent edge nodes, and it can be clearly seen that as the number of edge nodes increases, the system performance increases significantly, and does not degrade performance with the increase in concurrency. When there are 10 edge nodes and the concurrency reaches 2000, the performance of the edge computing node is stable at around 400. We can concluded that PEC can effectively improve the overall performance of the system by increasing the edge nodes.

4) Performance of File Transmission in Traditional and SDN Network: In this experiment, we compared the performance of file uploading with different file sizes and file number in the traditional TCP/IP network and the SDN network. Because in the IIoT network, there are large volume of IoT devices and each of them can only generate a little data once a time, we choose the file size in 1, 5, and 10 MB, and the file number ranges from 20 to 100. We can see in Fig. 9, with the file number increases in different file sizes, the transmitting time under TCP/IP and SDN network increasing correspondingly. What's more, almost all transmissions in the TCP/IP cost more time than in the SDN network. Consequently, we can find that data transmitting in the SDN network has a better performance than in the traditional network overall.

VI. Security Analysis

A. Authenticity

All IIoT devices are supposed to register in the CA to be authenticated when they first come into the system. What's more, the CA will verify whether a user is enrolled whenever the user calls a smart contract on the blockchain. Some of our designed contracts also read the caller's certifications to check whether the caller's identity matches the contract parameters. Thus, the system reaches authenticity.

B. Data Confidentiality

All IIoT data are encrypted under symmetric keys and outsourced to the third-party edge servers. Attackers cannot reveal any information about the data without the symmetric keys. To manage the symmetric keys securely, server A first encrypts the symmetric keys with it's public key and uploads them to the blockchain. Even if other malicious members obtain the ciphertexts on the blockchain, they cannot decrypt them without A's private key. When A decides to share its data with another user B, it runs the IBPRE and generates the reencrypted cipher for B by a proxy. The reencrypted cipher which stored in the blockchain can only be decrypted by B. Therefore, our system guarantees data confidentiality during the data storage and sharing process.

C. Integrity

Before outsourcing data to edge servers, the device will first compute the hash value of data. The hash value is then uploaded to the blockchain attached with encrypted keys. After decrypting the data with the crypto keys, users can confirm whether the data

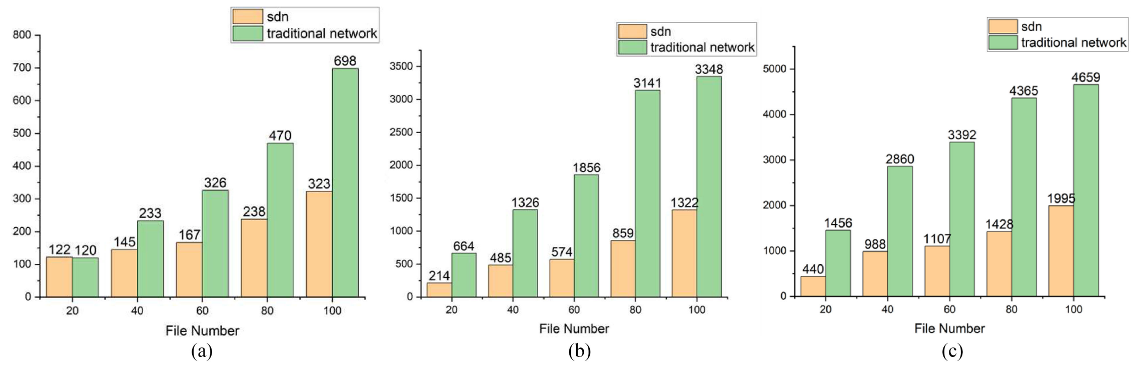

Fig. 9. File transmission performance in difference file size. (a) File Size = 1 MB. (b) File Size = 5 MB. (c) File Size = 10 MB.

have been modified by hashing the data again and comparing the new hash with the hash value from the blockchain.

VII. Conclusion

This article introduced a secure data sharing model for SDN-enabled PEC using blockchain and IBPRE. The SDN was implemented for flow forwarding locally. The encrypted devices data were outsourced to a third cloud server and the IBPRE scheme was adopted for securely sharing the cryptographic file keys between data owners and others. Users can upload crypto keys to the blockchain and interact with the blockchain, such as searching and updating records on it by designed smart contracts. Experiment results reveal that our proposed model has well performance and high throughput of the designed smart contracts.

References

[1] G. Aceto, V. Persico, and A. Pescapè, "A survey on information and communication technologies for Industry 4.0: State-of-the-art, taxonomies, perspectives, and challenges," *IEEE Commun. Surv. Tut.*, vol. 21, no. 4, pp. 3467–3501, Oct.–Dec. 2019.

[2] Y. Huang, J. Zhang, J. Duan, B. Xiao, F. Ye, and Y. Yang, "Resource allocation and consensus on edge blockchain in pervasive edge computing environments," in *Proc. IEEE 39th Int. Conf. Distrib. Comput. Syst.*, 2019, pp. 1476–1486.

[3] R. Roman, J. Lopez, and M. Mambo, "Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges," *Future Gener. Comput. Syst.*, vol. 78, pp. 680–698, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X16305635

[4] A. C. Baktir, A. Ozgovde, and C. Ersoy, "How can edge computing benefit from software-defined networking: A survey, use cases, and future directions," *IEEE Commun. Surv. Tut.*, vol. 19, no. 4, pp. 2359–2391, Oct.–Dec. 2017.

[5] I. Farris, T. Taleb, Y. Khettab, and J. Song, "A survey on emerging SDN and NFV security mechanisms for IoT systems," *IEEE Commun. Surv. Tut.*, vol. 21, no. 1, pp. 812–837, Jan.–Mar. 2019.

[6] D. E. Kouicem, A. Bouabdallah, and H. Lakhlef, "Internet of things security: A top-down survey," *Comput. Netw.*, vol. 141, pp. 199–221, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128618301208

[7] S. Nakamoto *et al.*, "Bitcoin: A peer-to-peer electronic cash system," 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[8] M. Blaze, G. Bleumer, and M. Strauss, "Divertible protocols and atomic proxy cryptography," in *Proc. Int. Conf. Theory Appl. Cryptographic Techn.*, 1998, pp. 127–144.

[9] F. Casino, T. K. Dasaklis, and C. Patsakis, "A systematic literature review of blockchain-based applications: Current status, classification and open issues," *Telemat. Informat.*, pp. 55–81, 2018.

[10] H. Wang, S. Ma, H.-N. Dai, M. Imran, and T. Wang, "Blockchain-based data privacy management with nudge theory in open banking," *Future Gener. Comput. Syst.*, vol. 110, pp. 812–823, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X18322702

[11] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," *ACM Trans. Inf. Syst. Secur.*, vol. 9, no. 1, pp. 1–30, 2006.

[12] M. Green and G. Ateniese, *Identity-Based Proxy Reencryption* (Series Applied Cryptography and Network Security). Berlin, Germany: Springer, pp. 288–306., 2006.

[13] Q. Wen, Y. Gao, Z. Chen, and D. Wu, "A blockchain-based data sharing scheme in the supply chain by IIoT," in *Proc. IEEE Int. Conf. Ind. Cyber Phys. Syst.*, 2019, pp. 695–700.

[14] J. Zhou, H.-N. Dai, and H. Wang, "Lightweight convolution neural networks for mobile edge computing in transportation cyber physical systems," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 6, Oct. 2019, Art. no. 67. [Online]. Available: https://doi.org/10.1145/3339308

[15] Z. Ning *et al.*, "Mobile edge computing enabled 5G health monitoring for internet of medical things: A decentralized game theoretic approach," *IEEE J. Sel. Areas Commun.*, pp. 1–16, 2020.

[16] Z. Ning *et al.*, "Intelligent edge computing in internet of vehicles: A joint computation offloading and caching solution," *IEEE Trans. Intell. Transp. Syst.*, to be published, doi: 10.1109/TITS.2020.2997832.

[17] S. Guo, X. Hu, S. Guo, X. Qiu, and F. Qi, "Blockchain meets edge computing: A distributed and trusted authentication system," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 1972–1983, Mar. 2020.

[18] M. Zhaofeng, W. Xiaochang, D. K. Jain, H. Khan, G. Hongmin, and W. Zhen, "A blockchain-based trusted data management scheme in edge computing," *IEEE Trans. Ind. Informat.*, vol. 16, no. 3, pp. 2013–2021, Mar. 2020.

[19] P. K. Sharma, M. Chen, and J. H. Park, "A software defined fog node based distributed blockchain cloud architecture for IoT," *IEEE Access*, vol. 6, pp. 115–124, 2018.

[20] K. Kataoka, S. Gangwar, and P. Podili, "Trust list: Internet-wide and distributed iot traffic management using blockchain and SDN," in *Proc. IEEE 4th World Forum Internet Things*, 2018, pp. 296–301.

[21] W. Jia, H. Zhu, Z. Cao, L. Wei, and L. Xiaodong, "SDSM: A secure data service mechanism in mobile cloud computing," in *Proc. IEEE Conf. Comput. Commun. Workshops*, 2011, pp. 1060–1065.

[22] M. Egorov, M. Wilkison, and D. Nuñez, "Nucypher KMS: Decentralized key management system," 2017. [Online]. Available: https://arxiv.org/abs/1707.06140

[23] N. McKeown *et al.*, "Openflow: Enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, 2008.

Ying Gao (Member, IEEE) received the B.S. degree in information engineering and the M.S. degree in computer application technology from the Central South University of China, Changsha, China, in 1997 and 2006, respectively, and the Ph.D. degree in computer application technology from the South China University of Technology, Guangzhou, China, in 2006.

She is currently a Professor with the School of Computer Science and Engineering, South China University of Technology. Her current research interests include service-oriented computing technology, software architecture, and blockchain and network security.

Yijian Chen received the B.S. degree in E-commerce from the South China University of Technology, Guangzhou, China, in 2018, where he is currently working toward the M.S. degree in computer science and engineering.

His current research interests include blockchain and PRE.

Xiping Hu (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from The University of British Columbia, Vancouver, BC, Canada, in 2015.

He is currently a Professor with the Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Beijing, China. He is the Co-Founder and Chief Scientist of Erudite Education Group in Hong Kong, a leading language learning mobile application company with more than 100 million users and listed as top two language education platform globally. His research interests are mobile CPS, crowd sensing, social networks, and cloud computing. He has authored or co-authored more than 100 papers published and presented in prestigious conferences and journals, such as the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING, the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the IEEE INTERNET OF THINGS JOURNAL, ACM TRANSACTIONS ON MULTIMEDIA COMPUTING, COMMUNICATIONS, AND APPLICATIONS, IEEE COMMUNICATIONS SURVEYS AND TUTORIALS, IEEE COMMUNICATIONS MAGAZINE, IEEE NETWORK, Hawaii International Conference on System Sciences, ACM MobiCom, WWW, and AAAI.

Prof. Hu has been serving as the Associate Editor for the IEEE ACCESS, and Lead Guest Editor for the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING.

Hongliang Lin received the B.S. degree in information security from South China University of Technology, Guangzhou, China, in 2018, where he is currently working toward the M.S. degree in computer science and engineering.

His current research interests include blockchain and its application.

Yangliang Liu received the B.S. degree in computer science and technology in 2018 from the South China University of Technology, Guangzhou, China, where he is currently working toward the M.S. degree in computer science and engineering.

His current research interests include blockchain technology and Internet of Things (IoT).

Laisen Nie received the Ph.D. degree in communication and information system from Northeastern University, Shenyang, China, in 2016.

He is currently an Associate Professor with the School of Electronics and Information, Northwestern Polytechnical University, Xi’an, China. His research interests include network measurement, network security, and cognitive networks.