Blockchain-Enabled Accountability Mechanism Against Information Leakage in Vertical Industry Services

Yang Xu, *Member, IEEE*, Cheng Zhang, *Student Member, IEEE*, Quanrun Zeng, Guojun Wang, *Member, IEEE*, Ju Ren, *Member, IEEE*, and Yaoxue Zhang, *Senior Member, IEEE*

***Abstract*—The emergence of 5 G technology contributes to create more open and efficient eco-systems for various vertical industries. Especially, it significantly improves the capabilities of the vertical industries focusing on content-sharing services like mobile telemedicine, etc. However, cyber threats such as information leakage or piracy are more likely to occur in an open 5 G networks. So tracking information leakage in 5 G environments has become a daunting task. The existing tracing and accountability schemes have nonnegligible limitations in practice due to the dependence on a Trusted Third Party (TTP) or being encumbered with the significant overhead. Fortunately, the blockchain helps to mitigate these problems. In this paper, we propose a blockchain-enabled accountability mechanism against information leakage in the content-sharing services of the vertical industry services. For any information converted to vector form, we use the blockchain technology to ensure that service providers and clients can securely and fairly generate and share watermarked content. Besides, the homomorphic encryption is introduced to avoid the disclosure of the watermarking content, which guarantees the subsequent TTP-free arbitration. Finally, we theoretically analyze the security of the scheme and verify its performance.**

***Index Terms*—Information leakage, blockchain, watermark, traceability, arbitration, vertical industry service.**

I. Introduction

The past decades have witnessed the rapid evolutions of computing paradigms and communication technologies. The emergences of network computing [1]–[3] and 5G-led future network techniques [4], [5] bring huge benefits to the human society and contribute a lot to create more open, flexible and efficient application eco-systems for multiple crucial vertical industries including healthcare, education, transportation, and governance [6]–[8]. By satisfying the critical requirements of low latency, high mobility and reliability in large-volume data transmissions, 5 G techniques significantly improve the capabilities and availabilities of various vertical industry services focusing on content-sharing services, such as mobile telemedicine and distance education [9].

However, cyber threats such as information leakage and piracy are still foreseen and become even more severe in future 5 G scenarios because the confidential information has more potential to be illegally downloaded, uploaded and spread in the open 5G-enabled Internet of Everything (IoE) environments, while the corresponding surveillance and tracking become very daunting tasks [10]. For example, the improper disclosure of confidential information regarding patients' privacy, such as the images of ultra-sound, X-ray and computed tomography (CT), may easily occur in content sharing services and can be quite difficult to track down the source of a leak in mobile telemedical systems built upon open 5 G networks [11], [12].

To mitigate the above information leakage problem in the content sharing services, the earlier scheme [13] attempts to embed the digital watermarking into the multimedia content such as CT to protect copyright, so that the honest data owner can trace the leaker according to the pirated copy. In these early approaches, the watermark is usually embedded by the data owner, which gives him obviously advantages and brings potential risks to users [14]. A malicious data owner can frame an honest data consumer by actively leaking the watermarked content and claim that it is done by the data consumer. To avoid this, some Buyer-Seller Watermarking (BSW) protocols [15]–[21] have already been proposed, in which the data owner as a service provider will produce watermarked content with the client and cannot know the complete watermark, thereby ensuring the fair embedding of the watermark. However, most BSW protocols introduce a Trusted Third Party (TTP) to generate the watermarked content, a centralized TTP may suffer the performance bottlenecks and single-point failures, which leads to difficulty in deployment in a vertical industry service environment. There are also some improved schemes [22]–[24] for reducing the dependence of watermarking embedding on TTP, in which a TTP is still required for the arbitration.

The rise of blockchain technology [25] has made it possible to solve the above problems. With the properties of the blockchain such as decentralization, tamper resistance, and traceability,

Manuscript received September 3, 2019; revised January 18, 2020; accepted February 21, 2020. Date of publication February 27, 2020; date of current version July 7, 2021. This work was supported by the Fundamental Research Funds for the Central Universities (No. 531118010454), in part by the National Natural Science Foundation of China (No. 61632009), and in part by the Guangdong Provincial Natural Science Foundation (No. 2017A030308006). Recommended for acceptance by Dr. M. Atiquzzaman. *(Corresponding author: Guojun Wang.)*

Yang Xu is with the College of Computer Science and Electronic Engineering, Hunan University, Changsha 410082, China (e-mail: xuyangcs@hnu.edu.cn).

Cheng Zhang, Quanrun Zeng, and Yaoxue Zhang are with the School of Computer Science and Engineering, Central South University, Changsha 410083, China (e-mail: zhangcheng_sy@csu.edu.cn; zengqquanrun@csu.edu.cn; zyx@csu.edu.cn).

Guojun Wang is with the School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou 510006, China (e-mail: csgjwang@gmail.com).

Ju Ren is with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China (e-mail: renju@csu.edu.cn).

Digital Object Identifier 10.1109/TNSE.2020.2976697

some schemes [26], [27] introduce the blockchain in to save content sharing records, but they lack the traceability and accountability of leaked content. Recently, some schemes [28], [29] combine digital watermarking and blockchain technologies for the traceability of the information leakage. For example, Bhowmik *et al.* [29] embed a digital watermark in the shared multimedia content, and recorded the watermark with the blockchain, thereby ensuring the consistency of the on-chain and off-chain information to achieve the traceability of information leakage. However, these schemes cannot prevent false accusation fraud by malicious service providers and cannot achieve fair dispute arbitration. Besides, the storage and computation overhead is also a huge challenge for these blockchain-based systems.

This paper proposes a new blockchain-enabled accountability mechanism against information leakage. For the information being distributed, this mechanism allows service providers to embed watermarking into private copies without the knowledge of the watermarking of corresponding clients, and also achieves the detection and arbitration of users who illegally disclosed information. In our scheme, the contents published by service providers are transformed into feature domain and encrypted by encryption Lookup Table (LUT). In the distribution phase, the service provider and the client secretly generate a fragment of the watermarking, and then the service provider figures out the client-specific personalized decryption LUT based on the homomorphic encryption public key system. Finally, the watermarking will be embedded in the corresponding content when the client decrypts using the decryption LUT. When the service provider finds that the content has been leaked illegally, he can extract a watermark with the client's identity information from the content and conduct arbitration based on the transaction information stored on the blockchain without the involving of the TTP. We prove the security and validity of the protocol itself under the basic assumption of the security of the homomorphic encryption system. In addition, the experiment shows that the scheme is feasible. In conclusion, the major contributions of our work are four-fold:

1) We propose a blockchain-based traceability mechanism in the sake of tracking the information leakage behavior in the multimedia content transactions. For any delivered private copy, a watermark is embedded through a smart contract-based delivering process to identify the corresponding private copyholder.
2) We design a TTP-free information leakage accountability mechanism based on watermarking technique and smart contract. The arbitration smart contract will arbitrate correctly according to the evidence submitted by both participants and the corresponding transaction records on the blockchain.
3) With the introduction of LUT-based security embedding and homomorphic encryption technologies, we proposed a secure client-side watermarking embedding mechanism for the traceability and fairness, in which the service provider and the client can jointly generate and embed the watermark, without the knowledge of complete watermark.
4) We analyze the security of the mechanism and establish a prototype system based on the Ethereum to test effectiveness and performance.

The rest part of this paper is organized as follows. Section II introduces some basic concepts about homomorphic encryption and LUT-based secure embedding technologies. Section III introduces the thread model. In Section IV, we propose a blockchain-based accountability mechanism and analyzed its security in Section V. Section VI verifies the effectiveness and performance of the approach by experiments. Section VII introduces some backgrounds and related works. Finally, the last section summaries this paper.

II. Preliminary

A. Homomorphic Encryption

Let $\mathrm{E}(\cdot)$ denotes the encryption operator which maps an element in a set A to the corresponding element in set B. For any $m_1, m_2 \in A$, encryption is called to be homomorphic if the following equation holds:

$$E(m_1) \odot E(m_2) = E(m_1 \oplus m_2) \tag{1}$$

In which $\oplus$ and $\odot$ are the operations in the ring $(A, \oplus)$ and $(B, \odot)$ respectively. If $\oplus$ denotes the multiplication defined, the encryption system is called multiplicative homomorphic encryption. Samely, such cryptosystem is called additive homomorphic encryption when it denotes the additive operations.

Homomorphic encryption allows a client to perform calculations in the ciphertext domain and obtain the encryption results of the corresponding plaintext domain calculations. This feature allows us to delegate computing to others and without revealing valuable information. The homomorphic encryption scheme used in this paper is Paillier [30], a probabilistic asymmetric algorithm and additive homomorphic which satisfying

$$D(E(m_1, r_1) \cdot E(m_2, r_2)) = m_1 + m_2 \tag{2}$$

in which r_1 and r_2 are random positive integers over a finite field. They are used to map the same plaintext into different ciphertext. In addition, if a is an integer, Paillier also satisfies

$$D(E(m_1, r_1)^a) = a \cdot m_1 \tag{3}$$

B. Lookup Table-Based Secure Embedding

The lookup table (LUT) based secure embedding, proposed by Celik *et al.* [31], can support the service provider to encrypt a digital content, and provide different decryption keys to different clients so that the personalized watermark is embedded when the client decrypts the content.

In detail, the original digital content can be represented as a feature vector $\mathbf{x} = (x_0, x_1, \ldots, x_{N-1})$ with the length N. The service provider generates a long-term LUT $\mathbf{E} = (E_0, E_1, \ldots, E_{J-1})$ to encrypt the content. The entries of $\mathbf{E}$ are chosen independently and randomly following a Gaussian distribution $\mathcal{N}(0, \sigma_E)$. The service provider also generates an $N \times R$ matrix $\mathbf{t} = \{t_{ij}\}$ in which $t_{ij} \in [0, J-1]$. The size of the security parameter R is positively correlated with the security of encryption. The service provider can encrypt his content to obtain the encrypted vector $\mathbf{y} = (y_0, y_1, \ldots, y_{N-1})$ according to the following equation.

$$y_i = x_i + \sum_{j=0}^{R-1} E_{t_{ij}} \tag{4}$$

Then the service provider generates a personalized watermark LUT $\mathbf{W}_{\mathrm{c}} = (W_0, W_1, \ldots, W_{J-1})$ whose entries follow $\mathcal{N}(0, \sigma_W)$ and a corresponding decryption LUT $\mathbf{D}_{\mathrm{c}} = (D_{\mathrm{c},0}, D_{\mathrm{c},1}, \ldots, D_{\mathrm{c},J-1})$ for a client. For $j \in \{0, 1, \ldots, J-1\}$, the $D_{\mathrm{c},j}$ is figured out as follows:

$$D_{\mathrm{c},j} = -E_j + W_{\mathrm{c},j} \tag{5}$$

The encrypted content and $\mathbf{D}_c$ will be sent to the client through a secure channel.

After receiving y and $\mathbf{D}_c$, the client can decrypt and get the watermarked content according to $\mathbf{D}_c$. The decryption of each y_i is as follows.

$$x_{c,i} = x_i + \sum_{j=0}^{R-1} \mathbf{D}_{\mathrm{c},t_{ij}} = y_i + \sum_{j=0}^{R-1} \mathbf{W}_{\mathrm{c},t_{ij}} = x_i + w_{\mathrm{c},i} \tag{6}$$

Finally, the watermark w_c is the sum of R entries of $\mathbf{W}_c$, and the client obtain the watermarked content $x_c = x + w_c$.

C. Blockchain Technology

The concept of blockchain was first appeared in the bitcoin project [25]. Blockchain is a distributed database that integrates technologies such as encryption, consensus mechanism and point-to-point network. The data on the blockchain is packaged into multiple data blocks and protected with a hash function, which is stored in a chain structure in all blockchain nodes. This distributed storage ensures that the blockchain is difficult to tamper with. By maintaining data blocks in a chain structure connected in chronological order, any data on the blockchain can be traced back. And the traceability makes the blockchain widely used in information supervision, data recovery, etc. However, to maintain the consistency of data among nodes that do not trust each other, a consensus mechanism is needed. At present, common consensus mechanisms include PoW (Proof of Workload), PoS (Proof of Stake) and PoA (Proof of Authority). Among them, the PoW mechanism is completely decentralized, and nodes determine the right to generate blocks through competition. While the PoA consensus mechanism uses some reputable auditors to cooperate and maintain the blockchain, which avoids multiple nodes competing for accounting rights, thus reducing the overhead of generating blocks. In addition to supporting digital cash, Ethereum [32] has further implemented smart contracts on the blockchain. A smart contract is an automatically executed digital contract and it is Turing computable. In the Ethereum ecosystem, smart contracts are run by all nodes in the blockchain network in their Ethereum Virtual Machine (EVM) and produce results.

TABLE I
NOTATIONS AND DEFINITIONS

Notation	Definition
$\mathrm{E_c}(\cdot, r)$	Encrypting a plaintext with Paillier.
r	a random positive integer over a finite field.
$\mathbf{r}$	a value formed by concatenating multiple random numbers with zero padding, $\mathbf{r} = r_0 \Vert r_1 \Vert \ldots \Vert r_{k_1}$.
$\mathrm{D_c}(\cdot)$	Decrypting a ciphertext with Paillier.
$\mathbf{M}_{\mathrm{right}}^{-1}$	The right inverse of the corresponding matrix $\mathbf{M}$, in which $\mathbf{M}_{\mathrm{right}}^{-1} = \mathbf{M}^{\mathrm{T}}(\mathbf{M}\mathbf{M}^{\mathrm{T}})^{-1}$.
H_r	a hash value of $\mathbf{r}$, $H_r = Hash(\mathbf{r})$.
$[\mathbf{x}]_{\mathrm{E_c}}$	The vector obtained by calculating Paillier encryption for each component x_i in a given vector $\mathbf{x}$.
R	a security parameter of the LUT-based embedding.
$\mathbf{b}_{\mathrm{c}}$	The binary fingerprint vector provided by a client c.
$\mathbf{b}_{\mathrm{s}}$	The binary fingerprint fragments vector provided by a service provider s.
$\mathbf{b}$	The complete binary fingerprint, $\mathbf{b} = \mathbf{b}_{\mathrm{s}} \Vert \mathbf{b}_{\mathrm{c}}$.
$\mathbf{m}_{\mathrm{c}}$	The encoded fingerprint fragments vector of $\mathbf{b}_{\mathrm{c}}$.
$\mathbf{m}_{\mathrm{s}}$	The encoded fingerprint fragments vector of $\mathbf{b}_{\mathrm{s}}$.
$\mathbf{m}$	The complete encoded fingerprint, $\mathbf{m} = \mathbf{m}_{\mathrm{s}} \Vert \mathbf{m}_{\mathrm{c}}$.
$\mathbf{w}$	The client-related watermark which will be embedded in a private copy, $\mathbf{w} = \mathbf{WT}$.
$\mathbf{x}$	A N-dimensional content feature vector.
$\mathbf{y}$	The ciphertext vector encrypted by $\mathbf{x}$ through LUT $\mathbf{E}$ and index matrix $\mathbf{T}$, in which $\mathbf{y} = \mathbf{x} + \mathbf{ET}$.
$\mathbf{x}_{\mathrm{c}}$	The feature vector of a private copy held by client c, which is embedded with a user-related watermark $\mathbf{w}$.
$\mathbf{E}$	The encryption LUT.
$\mathbf{W}_{\mathrm{c}}$	The watermark LUT of a private copy of a client.
$\mathbf{D}_{\mathrm{c}}$	The decryption LUT of a private copy of a client.
K	The length of the watermark embedded in the content.
$\mathbf{b}*$	The evidence extracted by the service provider from the private copy
$\mathbf{b}'$	The fingerprint uploaded by the client in the arbitration phase

III. Threat Model

We assume that the user may be malicious. They are motivated to maliciously disseminate private copies of the digital content they hold and attempt to escape punishment by deception in the ensuing arbitration phase. Besides, considering the common characteristics of digital content involved in this model, we assume that service providers will correctly provide the services required by users and execute the corresponding processes honestly.

On this basis, we reasonably assume that neither service providers nor users are motivated to take unreasonable actions that may harm their interests in the verification stage, that is, an innocent participant will not self-harm by providing false evidence. At the same time, in this scheme, As only one arbitration is conducted for the same user and the same digital content, the service provider will not launch false accusations against specific users without any reason in the uncertain situation.

Besides, we assume that the basic blockchain network and encryption technology are uncompromising to both verifiers and users, that is, the defense against attacks on blockchain infrastructure and attacks against encryption algorithm is out of our scope.

IV. Approach

In this section, we introduce the framework of our information leakage accountability mechanism and separately describe the watermark embedding mechanism and the arbitration process in detail. To facilitate the description, the notations used in this paper are listed in Table I.

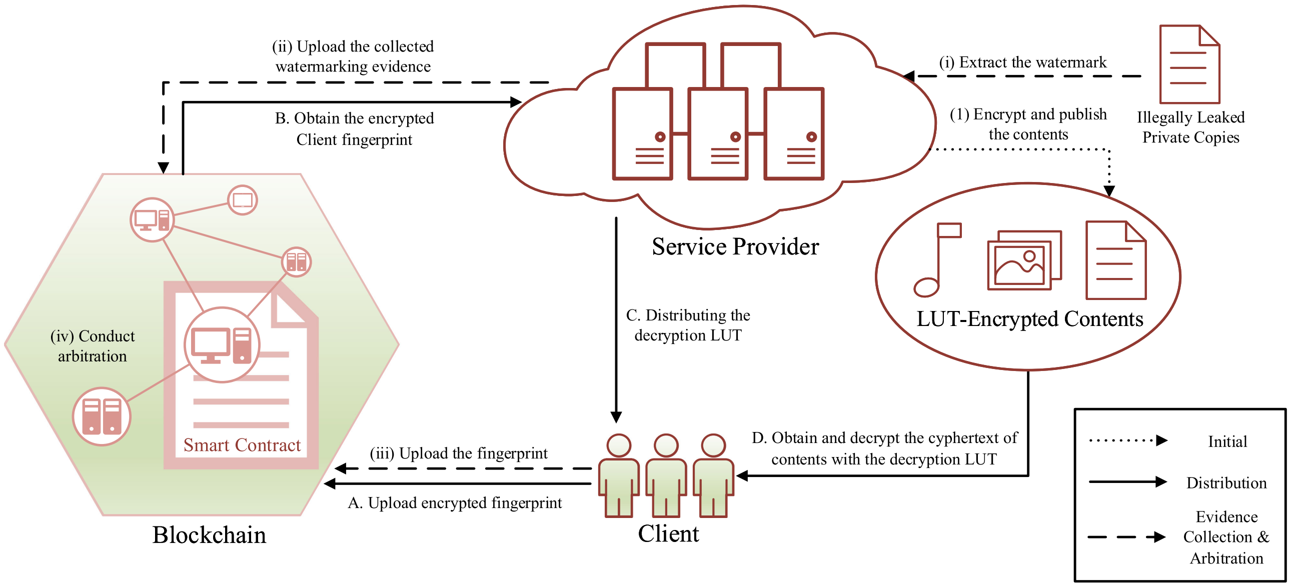

(ii) Upload the collected watermarking evidence
B. Obtain the encrypted Client fingerprint
(i) Extract the watermark
(1) Encrypt and publish the contents
Illegally Leaked Private Copies
Service Provider
C. Distributing the decryption LUT
LUT-Encrypted Contents
(iv) Conduct arbitration
Smart Contract
Blockchain
(iii) Upload the fingerprint
A. Upload encrypted fingerprint
Client
D. Obtain and decrypt the cyphertext of contents with the decryption LUT
Initial
Distribution
Evidence Collection & Arbitration

Fig. 1. Framework of the Blockchain-enabled Accountability Mechanism.

A. *Framework and Workflow*

As represented in Fig. 1, there are two main entities in the system, service provider and client. The service provider is the data owner and will provide clients with content sharing services. The client is a user of data and will obtain and use digital content from the service provider. Each entity is connected to the blockchain network. In addition, some nodes in the system are responsible for maintaining the blockchain and profiting from transaction fees.

By introducing blockchains and smart contract technology, our scheme implements an arbitrable information leakage tracking mechanism. In this mechanism, the service provider encrypts its digital contents based on LUT and publishes the ciphertext in the network. The client trying to access specific content will use a blockchain-based distribution mechanism to obtain a decryption LUT embedded with watermarking information, with which the client can decrypt a specific ciphertext and get the corresponding content embedded with watermarking. The complete scheme includes four phases: initial, distribution, evidence extraction, and arbitration. For any digital content that could be leaked, the service provider will encrypt this information with LUT before sending it out during the initial phase. When a client attempts to obtain the original information from encrypted content, it will submit a request to the service provider to initiate the distribution phase, in which the service provider will generate a specific decryption LUT for the client. The evidence extraction phase starts when the service provider finds a copy of private content illegally circulated, which will help the service provider identify the owner of the private copy and collect evidence. When necessary evidence is obtained, the service provider will initiate the arbitration phase through a smart contract deployed on the blockchain, and the smart contract will seek evidence from the indicted user and service provider and make judgments accordingly.

B. *Initial Phase*

For ease of description, we will describe the transaction watermarking protocol for a single consignment. The service provider first converts its data into vector form. Considering a content modulated as N-dimensional feature vector $\mathbf{x} = (x_0, x_1, \ldots, x_{N-1})$, $\mathbf{x}$ will be encrypted into a vector $\mathbf{y} = (y_0, y_1, \ldots, y_{N-1})$, which can be described as follows:

$$\mathbf{y} = \mathbf{x} + \mathbf{ET} \tag{7}$$

in which $\mathbf{E} = (E_0, E_1, \ldots, E_{J-1})$ is an encryption LUT generated by the service provider and E_j is i.i.d. random variables following a Gaussian distribution $\mathcal{N}(0, \sigma_E)$ for $j \in \{0, 1, \ldots, J-1\}$. $\mathbf{T} = \{t_{ij} | t_{ij} \in [0, R], 0 \leq i < N-1, 0 \leq j < J-1\}$ is a $J \times N$ matrix generated by the service provider. The security parameter R is set to an positive integer to prevent potential attacks. The sum of the elements of each row in $\mathbf{T}$ is R. Let $\mathbf{D} = -\mathbf{E}$, it is easily seen that $\mathbf{y}$ can be reduced to $\mathbf{x}$ by $\mathbf{D}$, as shown below.

$$\begin{aligned} \mathbf{x} &= \mathbf{y} - \mathbf{ET} \\ &= \mathbf{y} + \mathbf{DT} \end{aligned} \tag{8}$$

The service provider then releases $\mathbf{y}$ and its introduction, which are available for any client. When a client attempts to obtain $\mathbf{x}$, it will firstly download $\mathbf{y}$ and request $\mathbf{D}_{\mathrm{c}}$, with the embedded client-related watermark, from the service provider. The generation and distribution of $\mathbf{D}_{\mathrm{c}}$ in the distribution phase is described in the following subsection. Finally, the service provider will also establish smart

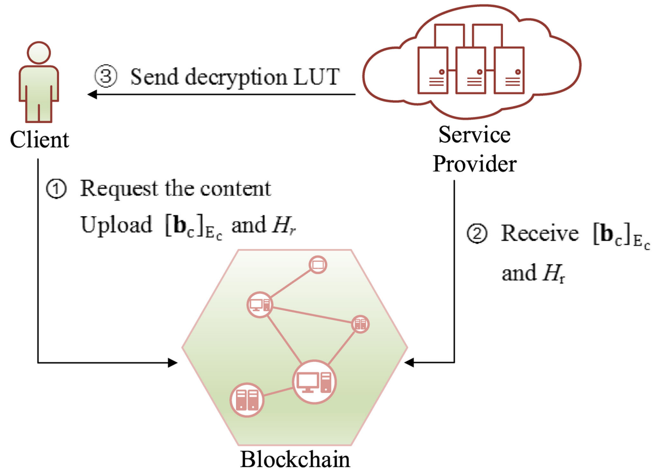

③ Send decryption LUT
Client
Service Provider
① Request the content
Upload $[\mathbf{b}_c]_{E_c}$ and H_r
② Receive $[\mathbf{b}_c]_{E_c}$ and H_r
Blockchain

Fig. 2. Workflow of the Distribution Phase.

contracts on the blockchain to conduct content transactions and arbitrate service disputes.

C. Distribution Phase

In the distribution phase, the service provider and the client will first determine the relevant parameters used in Paillier encryption. Then, as represented in Fig. 2, when the client attempts to access to the multimedia content, it will firstly secretly select a binary fingerprint vector $\mathbf{b}_c = (b_{c,0}, b_{c,1}, \ldots, b_{c,K_1-1})$, in which $b_{c,k} \in \{0, 1\}$ for any $k \in \{0, 1, \ldots, K_1 - 1\}$. Then the client will compute the corresponding encrypted vector $[\mathbf{b}_c]_{E_c}$ using Paillier as follows:

$$[\mathbf{b}_c]_{E_c} = \left(E_c(b_{c,0}, r_0), E_c(b_{c,1}, r_1), \ldots, E_c(b_{c,K_1-1}, r_{K_1-1})\right) \tag{9}$$

in which $r_0, r_1, \ldots, r_{k_1}$ are random selected positive integer (For readability, random numbers in subsequent encryption equations will be omitted). The client will then supplement these random numbers to a fixed size with zeros and concatenate to get the overall hash value $H_r = Hash(\mathbf{r}) = Hash(r_0||r_1||\ldots||r_{k_1})$. After that, the client will send $[\mathbf{b}_c]_{E_c}$ and H_r to the blockchain. While receiving $[\mathbf{b}_c]_{E_c}$, the service provider will acknowledge receipt on the blockchain. If the service provider refuses to confirm for some reason or fails to confirm within the specified time, the transaction will be canceled.

In order to perform LUT-based client-side secure embedding, the client's fingerprint needs to be encoded. Specifically, the service provider encode $\mathbf{b}_c$ into $\mathbf{m}_c = (m_{c,0}, m_{c,1}, \ldots, m_{c,K_1-1})$, in which

$$m_{c,k} = \begin{cases} \sigma_W & (b_{c,k} = 1) \\ -\sigma_W & (b_{c,k} = 0) \end{cases} . \tag{10}$$

As shown below, owing to the help of homomorphic encryption, the service provider can obtain the encrypted $\mathbf{m}_c$ with $[\mathbf{b}_c]_{E_c}$ instead of figure out the $\mathbf{m}_c$ directly.

$$\begin{aligned} E_c(m_{c,k}) &= E_c(2\sigma_W b_{c,k} - \sigma_W) \\ &= E_c(b_{c,k})^{2\sigma_W} E_c(-\sigma_W) \end{aligned} \tag{11}$$

In order to accurately locate the transaction corresponding to the leaked information, the service provider also needs to embed a fingerprint that can identify the transaction in the content. Assuming the watermark finally generated by the service provider is a K-dimensional vector $\mathbf{m}$, in a similar way as $\mathbf{b}_c$ and $[\mathbf{m}_c]_{E_c}$, the service provider will firstly generate a K_2-dimensional binary vector $\mathbf{b}_s$ and the corresponding $[\mathbf{m}_s]_{E_c}$, in which $K_2 = K - K_1$. Then the complete encoded fingerprint $\mathbf{m} = (m_0, m_1, \ldots, m_K)$ can be obtained as follows:

$$\mathbf{m} = \mathbf{m}_s||\mathbf{m}_c \tag{12}$$

in which

$$m_k = \begin{cases} m_{s,k} & (k \in [0, K_2)) \\ m_{c,k-K_2} & (k \in [K_2, K)). \end{cases} \tag{13}$$

Note that the above operations are based on the homomorphic public key cryptosystems, i.e.,

$$[\mathbf{m}]_{E_c} = [\mathbf{m}_s]_{E_c}||[\mathbf{m}_c]_{E_c} \tag{14}$$

After finishing the calculation of $[\mathbf{m}]_{E_c}$, the service provider will calculate the corresponding ciphertext of watermark LUT $\mathbf{W}_c = (W_{c,0}, W_{c,1}, \ldots, W_{c,J-1})$. Let $\mathbf{G} = \{G_{j,k}\}$ is a preset $K \times J$ matrix in order to mapping a K-bit fingerprint $\mathbf{m}$ into a J-dimensional LUT $\mathbf{W}_c$, i.e.

$$\mathbf{W}_c = \mathbf{m}\mathbf{G} \tag{15}$$

Therefore for $j \in \{0, 1, \ldots, J - 1\}$, the $E_c(W_j)$ can be obtained as follows:

$$\begin{aligned} E_c(W_{c,j}) &= E_c\left(\sum_{k=1}^{K-1} G_{j,k} \cdot m_k\right) \\ &= \prod_{k=0}^{K-1} E_c(m_k)^{G_{j,k}} \pmod n \end{aligned} \tag{16}$$

Finally, with the homomorphic encryption, the service provider embed $\mathbf{m}$ into the private decryption $\mathbf{D}_c$ in the encryption domain as follows:

$$\begin{aligned} E_c(D_{c,j}) &= E_c(D_j + W_{c,j}) \\ &= E_c(D_j) \cdot E_c(W_{c,j}) \end{aligned} \tag{17}$$

Then the encrypted LUT $[\mathbf{D}_c]_{E_c}$ will be sent to the client and be decrypted into $\mathbf{D}_c$. With $\mathbf{D}_c$ and the encrypted content feature $\mathbf{y}$, the client will obtained the watermarked private copy $\mathbf{x}_c$ as presented below:

$$\mathbf{x}_c = \mathbf{y} + \mathbf{D}_c\mathbf{T} \tag{18}$$

D. Evidence Collection Phase

When private copies of illegal circulation are obtained, the service provider will extract watermarks from the privacy copy through the evidence extraction mechanism and find the corresponding copy owner. Note that

$$\begin{aligned}\mathbf{x}_c &= \mathbf{y} + \mathbf{D}_{\mathrm{c}}\mathbf{T}\\ &= \mathbf{y} + (\mathbf{D}\mathbf{T} + \mathbf{W}_{\mathrm{c}}\mathbf{T})\\ &= \mathbf{x} + \mathbf{W}_{\mathrm{c}}\mathbf{T}\end{aligned} \tag{19}$$

Then we have

$$\mathbf{W}_{\mathrm{c}} = (\mathbf{x}_c - \mathbf{x})\mathbf{T}^{-1}_{\text{right}} \tag{20}$$

in which $\mathbf{T}^{-1}_{\text{right}} = \mathbf{T}^{\mathrm{T}}(\mathbf{T}\mathbf{T}^{\mathrm{T}})^{-1}$ is the right inverse of $\mathbf{T}$. Therefore the encoded fingerprint $\mathbf{m}^*$ embedded in $\mathbf{D}_{\mathrm{c}}$ can be expressed as

$$\mathbf{m}^* = (\mathbf{x}_{\mathrm{c}} - \mathbf{x})\mathbf{T}^{-1}_{\text{right}}\mathbf{G}^{-1}_{\text{right}} \tag{21}$$

Letting $\mathbf{b}^* = (b^*_0, b^*_1, \ldots, b^*_{K-1})$ is the watermark to be extracted. For $k \in \{0, 1, \ldots, K-1\}$, by maximum likelihood estimation, the extraction of b^*_k can be described as

$$b^*_k = \begin{cases} 0 & (m^*_k < 0)\\ 1 & (m^*_k \geq 0)\end{cases} \tag{22}$$

Within $\mathbf{b}^*$, the service provider can easily obtain $\mathbf{b}^*_{\mathrm{s}}$ and $\mathbf{b}^*_{\mathrm{c}}$ through the equation (13) as follows:

$$\begin{cases}\mathbf{b}^*_{\mathrm{s}} = (b^*_0, b^*_1, \ldots, b^*_{K_2-1})\\ \mathbf{b}^*_{\mathrm{c}} = (b^*_{K_2}, b^*_{K_2+1}, \ldots, b^*_{K-1}).\end{cases} \tag{23}$$

The service provider is able to determine the owner of the private copy with $\mathbf{b}^*_s$ and then initiate an arbitration request to the arbitration smart contract.

E. Arbitration Phase

After the process of the evidence collection phase, the service provider will initiate an arbitration request to the arbitration smart contract with the obtained $\mathbf{b}^*_{\mathrm{c}}$ and the identification information of the specific client.

The process of dispute arbitration is shown in Fig. 3. After receiving the arbitration request, the smart contract will notify the client to participate in arbitration. The client will first publish the random number $\mathbf{r}'$ used for their encrypted fingerprints on the blockchain. Note that the $\mathbf{r}'$ uploaded by the client during the arbitration phase may not be equal to the $\mathbf{r}$ uploaded during the distribution phase due to accident or malicious intent. Therefore, the smart contract will check whether the $\mathbf{r}'$ uploaded by the client is satisfied

$$H_r = Hash(\mathbf{r}') = Hash(r'_0||r'_1||\ldots||r'_{K_1-1}) \tag{24}$$

If the equation does not hold, which means that the client provides false $\mathbf{r}'$ in the arbitration phase, the smart contract will determine that the client has malicious behavior. On the contrary, the client will be asked to reveal $\mathbf{b}'_{\mathrm{c}}$. Similarly, the smart contract will continue to check whether the $\mathbf{b}'_{\mathrm{c}}$ uploaded by the client satisfies

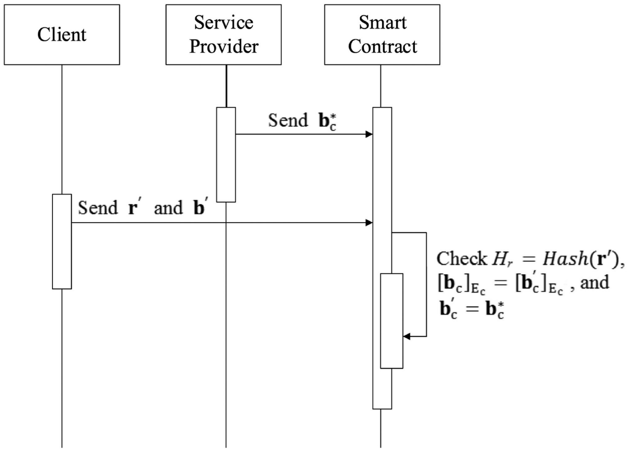

Client
Service Provider
Smart Contract
Send $\mathbf{b}^*_{\mathrm{c}}$
Send $\mathbf{r}'$ and $\mathbf{b}'$
Check $H_r = Hash(\mathbf{r}')$,
$[\mathbf{b}_{\mathrm{c}}]_{\mathrm{E_c}} = [\mathbf{b}'_{\mathrm{c}}]_{\mathrm{E_c}}$, and
$\mathbf{b}'_{\mathrm{c}} = \mathbf{b}^*_{\mathrm{c}}$

Fig. 3. Sequence Diagram of Arbitration Phase.

$$[\mathbf{b}'_{\mathrm{c}}]_{\mathrm{E_c}} = [\mathbf{b}_{\mathrm{c}}]_{\mathrm{E_c}} \tag{25}$$

Similarly, if $[\mathbf{b}'_{\mathrm{c}}]_{\mathrm{E_c}} = [\mathbf{b}_{\mathrm{c}}]_{\mathrm{E_c}}$, the smart contract will determine that the client has malicious behavior and attempts to escape arbitration. Otherwise, when equation (24) and (25) hold, the smart contract will then check whether $\mathbf{b}' = \mathbf{b}^*$. When $\mathbf{b}' = \mathbf{b}^*$ hold, which means that the private copy of the client has been circulated irregularly, the smart contract will judge it to be malicious. If $\mathbf{b}' \neq \mathbf{b}^*$, the $\mathbf{b}^*$ provided by the service provider is not sufficient to convict the client. Finally, the smart contract will automatically fine the malicious client or service provider based on the arbitration result.

V. Security Analysis

In this section, we assume that most nodes in the blockchain are trustworthy, which is easily satisfied in most blockchain application environments. The blockchain technology, Paillier encryption technology and LUT-based secure embedding technology used in the approach are reliable. Based on this assumption, we analyze the security of our scheme.

In our protocol, data such as $[\mathbf{b}_{\mathrm{c}}]_{\mathrm{E_c}}$ and H_r are published in blockchain as transactions. Blockchains adopt digital signature technology to ensure the integrity and security of data in the process of network transmission. Unless the adversary obtains the private key of a participant, it can not disguise other participants to tamper with the transaction information. Based on the hypothesis of the above threat model, we assume that the data stored in the blockchain is trustworthy and that the tampering of the data already published on the blockchain is beyond our consideration.

In addition, only the specified blockchain account in this scenario can call the function in the corresponding smart contract. When a user attempts to acquire digital content and initiates an application to the service provider, the smart contract will record the blockchain addresses of both parties, and only those addresses can invoke the corresponding functions. This setting ensures that only the relevant nodes can participate in the service process, and an attacker cannot invoke smart contracts to falsify data. That is, the data and smart contracts stored on the blockchain are trustworthy.

We first analyze the malicious behavior of the service provider. As mentioned in our threat model, a service provider will honestly execute the service protocol for its own interest. However, a malicious service provider may attempt to unconventionally obtain the client fingerprint fragments $\mathbf{b}_{\mathrm{c}}$ for the false accusation of the client, i.e., a service provider may attempt to figure out $\mathbf{b}_{\mathrm{c}}$ through H_r and $[\mathbf{b}_{\mathrm{c}}]_{\mathrm{E_c}}$ rather than the corresponding private copy $\mathbf{y}_{\mathrm{c}}$. However, considering the assumption on public key cryptosystem in our protocol, it is difficult to figure out $\mathbf{b}_{\mathrm{c}}$ with $[\mathbf{b}_{\mathrm{c}}]_{\mathrm{E_c}}$ and H_r, i.e., for the service provider, there is no polynomial time algorithm to solve $\mathbf{b}'_{\mathrm{c}}$ and $\mathbf{r}'$ which satisfies $[\mathbf{b}_{\mathrm{c}}]_{\mathrm{E_c}} = [\mathbf{b}'_{\mathrm{c}}]_{\mathrm{E_c}}$ and $H_r = Hash(\mathbf{r}')$. Therefore, the service provider cannot figure out the fingerprint fragments $\mathbf{b}_{\mathrm{c}}$ of client unless it obtains a private copy $\mathbf{y}_{\mathrm{c}}$.

Then we analyze the malicious behavior of the client. For a single arbitration, there are following possible cases of malicious client:

1) First, if a client tries to upload a false $\mathbf{r}'$ in the arbitration phase, due to the checking of $H_r = Hash(\mathbf{r}')$, it has to find a pair of $\mathbf{r}$ and $\mathbf{r}'$ satisfying $\mathbf{r} \neq \mathbf{r}'$ and $H_r = Hash(\mathbf{r}')$ in order to deceive the smart contract, which is difficult according to the properties of hash functions.
2) If the $\mathbf{r}'$ provided in the arbitration phase is false but the H_r provided in the distribution phase is genuine. Similar to the case (1), it can be detected easily by checking whether $H_r = Hash(\mathbf{r})$.
3) When a client tries to upload a false $\mathbf{b}'_{\mathrm{c}}$ in the arbitration phase, it needs to find a pair of $\mathbf{b}$ and $\mathbf{b}'$ satisfying $[\mathbf{b}_{\mathrm{c}}]_{\mathrm{E_c}} = [\mathbf{b}'_{\mathrm{c}}]_{\mathrm{E_c}}$ to pass the test (Note that the $\mathbf{r}'$ uploaded by the client has been verified at this time). Considering our assumptions about the involved encryption schemes, it is difficult for a malicious client to find such a pair of $\mathbf{b}_{\mathrm{c}}$ and $\mathbf{b}'_{\mathrm{c}}$.
4) If both the $\mathbf{r}'$ and H_r uploaded by the client are incorrect, it is still difficult for the client to find an $\mathbf{r}'$ such that each $\mathrm{E_c}(b_{\mathrm{c},i}, r'_i)$ is equal to each corresponding item in $[\mathbf{b}_{\mathrm{c}}]_{\mathrm{E_c}}$.
5) If both the $\mathbf{r}'$, H_r and $\mathbf{b}'_{\mathrm{c}}$ uploaded by client are false. A successful deception requires the client to find a pair of $\mathbf{r},' \mathbf{r}, \mathbf{b}'_{\mathrm{c}}, \mathbf{b}_{\mathrm{c}}$ satisfying $\mathbf{r}' \neq \mathbf{r}$, $\mathbf{b}'_{\mathrm{c}} \neq \mathbf{b}_{\mathrm{c}}$, $Hash(\mathbf{r}') = Hash(\mathbf{r})$ and $\mathrm{E_c}(b'_{\mathrm{c},i}, r'_i) = \mathrm{E_c}(b_{\mathrm{c},i}, r_i)$ where $b'_{\mathrm{c},i}$ is an item in $\mathbf{b}'_{\mathrm{c}}$. Considering our assumptions about encryption schemes, it is difficult for the client to find the satisfied $\mathbf{r}', \mathbf{r}, \mathbf{b}'_{\mathrm{c}}, \mathbf{b}_{\mathrm{c}}$.

In conclusion, according to our assumptions, that the blockchain, Paillier encryption and LUT-based embedding are secure, the blockchain can correctly save data uploaded by clients during the distribution phase, and can perform correct calculations during the arbitration phase. Paillier encryption ensures that the service provider cannot obtain client's fingerprint illegally. The LUT secure embedding technology ensures that the watermark can be correctly embedded in the content when the client decrypts the content and cannot be identified by the client. In addition, because Paillier encryption is difficult to crack and the hash function is collision-resistant, any illegal input by the client or service provider during the arbitration phase will be identified. When the smart contract completes the verifying of $H_r = Hash(\mathbf{r})$, $[\mathbf{b}_{\mathrm{c}}]_{\mathrm{E_c}} = [\mathbf{b}'_{\mathrm{c}}]_{\mathrm{E_c}}$ and $\mathbf{b}_{\mathrm{c}} = \mathbf{b}'_{\mathrm{c}}$, it can accurately determine whether the client leaks its private copy illegally. Therefore, our scheme ensures the correctness of arbitration without the participation of TTP.

TABLE II
SPECIFICATIONS OF DEVICES

Parameter	Service provider	Client	Blockchain node
CPU	2.4 GHz	2.2 GHz	2.2 GHz
RAM	16 GB	6 GB	16 GBB
Storage	1 TB	128 GB	512 GB
Network	300 Mb	100 Mb	300 Mb
OS	Ubuntu 16.04.3 LTS	CentOS 7.2-1511	MacOS 10.0

VI. Evaluation

In this section, we deployed a prototype system and conducted a series of experiments to evaluate the effectiveness and performance of the blockchain-enabled information leakage accountability mechanism.

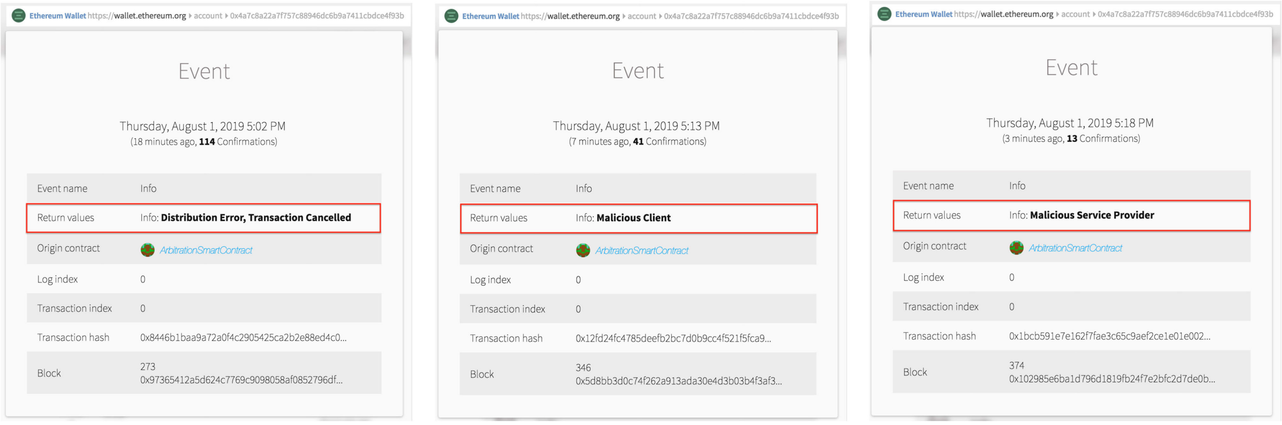
A. Evaluation Settings

In the experiment, a prototype system was built in the Ethereum environment as a proof of concept. The on-chain part of the system consists of smart contracts, and the off-chain part includes Paillier encryption, watermark embedding and extraction, etc. The program languages used include java, python, and solidity. Based on the Go implementation of the Ethereum protocol Geth 1.9.0, we constructed a consortium blockchain using the PoA consensus mechanism, and constructed a private blockchain using the PoW consensus mechanism for comparison. We set up 15 blockchain nodes to maintain the blockchain system and process transactions. The experiment simulates a scenario where 10 clients request digital content from 2 service providers to test the effectiveness and performance of the blockchain system. We let the blockchain node be the full node of the blockchain and store the complete blockchain information. The other entities are some light nodes, and only need to store some necessary information. The specific configuration of them is shown in Table. II.

For the effectiveness and performance of the watermark embedding and homomorphic encryption parts in the scheme, we used 50 8-bit gray images of different sizes to form the experimental data set. For each image, we map it to a vector using a Discrete Fourier Transform (DFT). We set the size of the LUT to 2^{16} and the size of R to 4. The entries of LUT are i.i.d. random variables following a Gaussian distribution. The homomorphic encryption algorithm used in the experiment is Paillier and its security parameters have 512 bits. We also set the length of the digital watermark to 64 bits and let the client and the service provider generate 32 bits respectively.

B. Evaluation Results

1) Effectiveness: In this section, we performed some experiments to test whether our system can correctly execute the protocol and properly arbitrate. To this end, we simulated



Ethereum Wallet https://wallet.ethereum.org ▸ account ▸ 0x4a7c8a22a7f757c88946dc6b9a7411cbdce4f93b

Event

Thursday, August 1, 2019 5:02 PM
(18 minutes ago, **114** Confirmations)

Event name	Info
Return values	Info: **Distribution Error, Transaction Cancelled**
Origin contract	ArbitrationSmartContract
Log index	0
Transaction index	0
Transaction hash	0x8446b1baa9a72a0f4c2905425ca2b2e88ed4c0...
Block	273 0x97365412a5d624c7769c9098058af0852796df...

(a) Transaction Cancelled

Ethereum Wallet https://wallet.ethereum.org ▸ account ▸ 0x4a7c8a22a7f757c88946dc6b9a7411cbdce4f93b

Event

Thursday, August 1, 2019 5:13 PM
(7 minutes ago, **41** Confirmations)

Event name	Info
Return values	Info: **Malicious Client**
Origin contract	ArbitrationSmartContract
Log index	0
Transaction index	0
Transaction hash	0x12fd24fc4785deefb2bc7d0b9cc4f521f5fca9...
Block	346 0x5d8bb3d0c74f262a913ada30e4d3b03b4f3af3...

(b) Malicious Client

Ethereum Wallet https://wallet.ethereum.org ▸ account ▸ 0x4a7c8a22a7f757c88946dc6b9a7411cbdce4f93b

Event

Thursday, August 1, 2019 5:18 PM
(3 minutes ago, **13** Confirmations)

Event name	Info
Return values	Info: **Malicious Service Provider**
Origin contract	ArbitrationSmartContract
Log index	0
Transaction index	0
Transaction hash	0x1bcb591e7e162f7fae3c65c9aef2ce1e01e002...
Block	374 0x102985e6ba1d796d1819fb24f7e2bfc2d7de0b...

(c) Malicious Service Provider

Fig. 4. Effectiveness Tests.

all possible behaviors of the client and the service provider. Based on previous analysis and assumptions, in addition to the correct behavior as specified in the scheme, there are also some behaviors that may lead to disputes as follows[1].

i) In the initial phase, a service provider may not confirm that he has receipted $[\mathbf{b}_{\mathrm{c}}]_{\mathrm{E_c}}$ and H_r for some reasons.

ii) After the distribution phase, a malicious client leaked the watermarked digital content.

iii) A malicious service provider may frame an honest client and apply for arbitration to claim the compensation.

iv) In the arbitration phase, the client may reveal a wrong $\mathbf{r}'$ or a wrong $\mathbf{b}'$ to deceive the arbitration smart contract.

In the experiment, we set the above scenarios to appear randomly to examine the effectiveness. Some typical experimental results are shown in Fig. 4.

Fig. 4(a) shows the result of the smart contract when the case (i) occurs. When the client uploads the encrypted fingerprint, the smart contract records the time the client uploaded the information, so when the smart contract is triggered again it can find that the service provider failed to confirm in time. Thus the transaction was terminated according to the plan.

When the case (ii) occurs, the service provider finds that the content is leaked. It finds the suspected client by extracting the watermark from the leaked content and applies for arbitration from the smart contract. If the arbitration proceeds smoothly, as shown in Fig. 4(b), the smart contract will confirm that the watermark belongs to the client, thereby determining that the client has maliciously leaked the content. In addition, when case (ii) and case (iv) occur simultaneously, as mentioned in the security analysis, since the hash function and encryption scheme we used are sufficiently secure, the client cannot find an $\mathbf{r}' \neq \mathbf{r}$ that satisfies $Hash(\mathbf{r}') = H_r$ and $[\mathbf{b}'_{\mathrm{c}}]_{\mathrm{E_c}} = [\mathbf{b}_{\mathrm{c}}]_{\mathrm{E_c}}$. Therefore, in the arbitration phase, the smart contract can find that the hash values do not match, and determine that the client is malicious. Similarly, when the case (iii) occurs, according to the threat model, the service provider cannot obtain the watermark during the watermark embedding process. Therefore, the fake $\mathbf{w}'$ constructed by the service provider cannot pass the examination of the smart contract. As shown in Fig. 4(c), the smart contract recognizes that the service provider is malicious.

In conclusion, experimental results prove that the security analysis is correct. Our system is able to terminate abnormal services in a timely manner, effectively discovering various malicious behaviors and conducting the arbitration.

2) Performance: In this subsection, we tested the gas consumption, throughput, and bloating rate of the blockchain to evaluate the performance of the on-chain part. We also tested the execution time of the off-chain part and demonstrated the watermark embedding process.

a) Gas Consumption. In the Ethereum ecosystem, smart contracts need to run in the Ethereum virtual machine (EVM), and every operation executed by EVM needs to consume a certain amount of gas. Therefore, the gas consumption reflects the computational overhead to execute an operation on the blockchain. In the scheme, there are four operations need to be performed on the blockchain. (1) The client uploads the encrypted fingerprint during the distribution phase. (2) The service provider receives and confirms the fingerprint. (3) During the arbitration phase, the service provider uploads the collected evidence. (4) The client uploads $\mathbf{b}'$ and $\mathbf{r}$ to trigger the smart contract for arbitration. We split the arbitration process into three steps to show gas consumption more clearly. The experimental results are shown in Fig. 5. Experimental results show that the gas consumption of operations a, b and c is quite small and does not exceed 60,000 units of gas. These three operations do not require the smart contract to perform complex calculations, and only need to save some data on the blockchain. So the gas consumption of these operations is proportional to the size of the data they need to save. The operations d, e and f jointly realize the arbitration function of the scheme. In addition to saving the data uploaded by the client, operation d also verifies the correctness of $\mathbf{r}$, and the total gas consumption does not exceed 60,000. The main operation in e is to perform a homomorphic encryption operation based on the information on the blockchain to verify the correctness of the data uploaded by the client, which consumes about

[1] Attacks on the watermark protocol or the encryption algorithm are out of our consideration. For example, we do not consider the case where encrypted watermark is cracked by a malicious service provider.

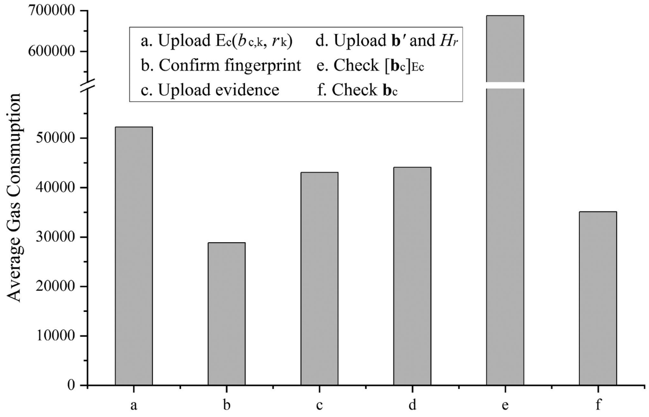


Fig. 5. The Average Gas Consumption.

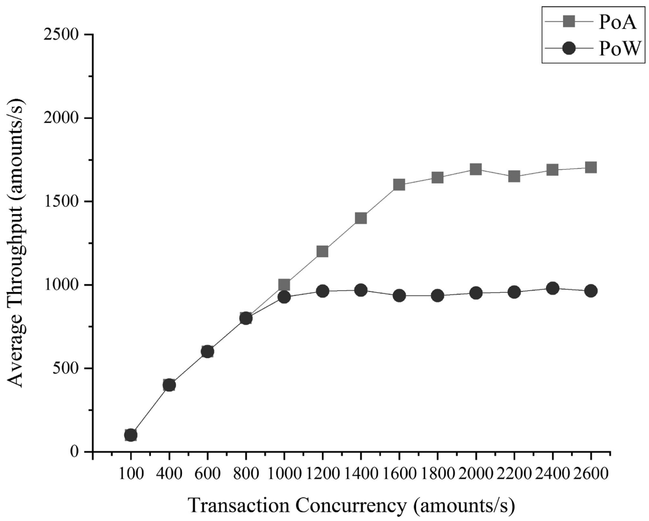


Fig. 6. The Average Throughput.

680,000 units of gas. Finally, operation f compares $\mathbf{b}'$ to $\mathbf{b}$, consuming about 35,000 units of gas. The experimental results show that our scheme has a high efficiency of executing the smart contract is acceptable to both clients and service providers, so our blockchain system is feasible in practice.

b) Transactions Throughput. The transactions throughput can reflect the capability of the blockchain system to handle concurrent transactions. Therefore, we examined the transaction throughput of our scheme under the PoA consensus mechanisms. We also deployed a blockchain system under the PoW consensus mechanism and conducted tests as a comparison.

As shown in Fig. 6, with our scheme based on the PoA consensus mechanism, the average throughput steadily increases in pace with the growth of concurrent transactions, and gradually work up to a stable peak at about 1650 times per second. As a contrast, the throughput curve of the PoW consensus mechanism-based scheme reaches to peak much faster than the PoA consensus mechanism, but its maximum throughput is only about half of the PoA based system. The reason is that the PoA consensus mechanism selects some more credible nodes to set up a signer voting committee to handle blockchain transactions. While the PoW consensus mechanism needs to waste a lot of computing resources. Therefore, our blockchain-based system has sufficient capacity to handle large volumes of transactions using the PoA consensus mechanism and is more suitable in a distributed environment.

TABLE III
BLOCKCHAIN BLOATING RATE

Transaction Rate	Per Minute	Per Hour	Per Day
500 transactions/min	0.015 MB	0.877 MB	21.04 MB
1000 transactions/min	0.029 MB	1.753 MB	42.081 MB
1500 transactions/min	0.044 MB	2.63 MB	63.121 MB
2000 transactions/min	0.058 KB	3.507 MB	84.161 MB

c) Bloating Rate. The bloating rate of the blockchain reflects the growth rate of the blockchain size as the number of transactions increases, which is inversely proportional to the scalability of the system. In our blockchain-based approach, each entity communicates with the other through the blockchain transactions. So the bloating rate also reflects the communication overhead of the system.

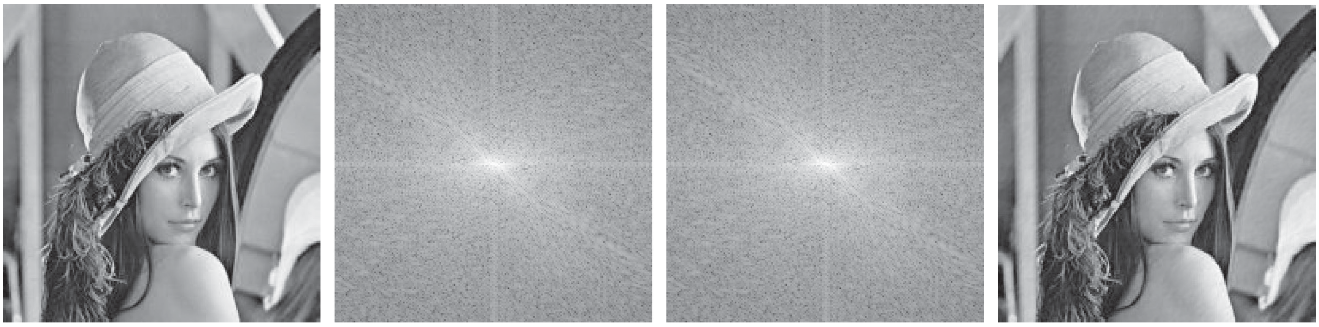
As shown in Table III, we tested the blockchain bloating rate for different transaction frequencies. When the blockchain-based system processes 500 transactions per minute, the size of the blockchain increases by 0.015 MB for a minute, 0.877 MB for an hour and 21.04 MB for a day. In this case, which is much higher than the actual transaction concurrency, the bloating rate of the prototype system is still acceptable for blockchain nodes. For light nodes in the blockchain such as clients and service providers, they only need to save the block header locally of each block, and the storage overhead of them is much smaller than authority nodes. Therefore, our system is feasible and acceptable.

d) Watermarking Scheme Performance. In this experiment, we tested the execution time of each step in the watermark embedding and extraction process under different conditions, and demonstrate the effectiveness of the watermarking scheme by showing the process of embedding a watermark into image content.

In the scheme, the client needs to generate a 32-bit watermark and encrypt it using Parllier encryption scheme. Then it also needs to perform the decryption operation on $[\mathbf{D}_c]_{E_c}$, the decryption operation on $\mathbf{y}$, and a transformation operation to obtain the watermarked image. The execution time of various operations at different image sizes are shown in Table IV. For a 2^{16} size LUT, the service provider can process a 1024 $\times 1024$ size picture in about 860 seconds, and the client can get watermarked content in 220 seconds. This experimental result is superior to the similar pixelwise approach and can be further optimized [22]. Therefore, this LUT-based watermark embedding technology has good performance that can effectively embed the watermark into the digital content in a short time, which can meet the needs of our system. In addition, Fig. 7 shows the process of transforming an image during the watermark embedding. It can be seen that the watermarked image is visually indistinguishable from the standard original image.

TABLE IV
EXECUTION TIME (IN SECONDS)

Size	Encrypting content	Encrypting fingerprint	Generating $[\mathbf{D}_c]_{E_c}$	Decrypting content	Extracting watermark
200 × 200	0.3059	0.0423	829.0448	231.5832	0.0004
521 × 512	1.9153	0.0414	822.4728	211.5286	0.0028
1024 × 1024	7.8433	0.0417	858.3247	212.061	0.0096



(a) Original Image (b) Original Image Spectrum (c) Watermarked Image Spectrum (d) Watermarked Image

Fig. 7. Watermark Embedding Process.

VII. RELATED WORK

At present, many schemes have been proposed to prevent data leakage and protect privacy in different environments [33], [34]. Among them, to trace the illegal leakage of content in a large-scale content distribution environment, some researches add an invisible digital watermark representing the client's identity to the content. However, a malicious service provider can frame a client if the watermark is directly embedded by himself. To prevent this attack, Menmon *et al.* [15] creatively proposed a buyer-seller watermarking (BSW) protocol in which the watermark is generated by a TTP and encrypted with the client's public key via a homomorphic public-key encryption scheme, so that the service provider can use the homomorphism for watermark embedding and without knowing the watermark information. Subsequently, some schemes that improved and extended the BSW protocol have been proposed [16]–[19]. Typically, Katzenbeisser *et al.* [20] extended BSW protocol by introducing the secure watermark embedding technology to reduce the overhead of computation and bandwidth. The secure watermark embedding technology can encrypt the multimedia content, and leave a watermark in the content during the decryption process through some special decryption methods [31]. These schemes do not need to perform homomorphic encryption operation to the content, so the computational overhead is greatly reduced. Besides, Peng *et al.* [21] introduce cloud computing to reduce the overhead of the service provider. However, the above scheme relies on a TTP to embed watermarks and resolve disputes, which exposes the system to centralized problems such as single points of failure and performance bottlenecks, and also may not exist in vertical industries.

Recently, there are also some non-TTP-based approaches have been proposed. Mina Dang *et al.* [22] implemented a BSW protocol that does not require TTP intervention when inserting the watermark. In their scheme, the service provider and the client respectively provide a partial watermark and embed it in the content using homomorphic encryption. With the introduction of RSA based homomorphic cryptosystem, Chen *et al.* [23] implemented a verifiable BSW protocol, in which the service provider can verify the correctness of the encrypted watermark generated by the client without TTP help. In addition, Bianchi and Piva [24] proposed a TTP-free asymmetric fingerprinting protocol using the client-side embedding technique. However, these schemes only improve the watermark generation mechanism but did not provide an effective arbitration mechanism. A TTP is still needed to help with arbitration when there is a data breach dispute between the service provider and client.

Since the nature of blockchain is suitable for tracking information leakages, recently it has been introduced in many approaches. Wang *et al.* [26] given a blockchain-based data tracing scheme, in which the blockchain is used to store encrypted content and record content distribution information for traceability of the data flow path. Similarly, Neisse *et al.* [27] provided a blockchain platform for data accountability and provenance tracking. However, since the content in these schemes does not contain any information about the owner when the content is illegally leaked, it is impossible to find the leaker based on the leaked content. Therefore, they cannot effectively solve the problem of information leakage in practice.

Meanwhile, some schemes that combine digital watermarking with blockchain technology to trace the information leakage have been proposed. Bhowmik and Feng [29] proposed a watermark-based multimedia blockchain framework. It records the content distribution information on the blockchain and embeds a watermark that points to the blockchain transaction address of the above record in the content. Meng *et al.* [28] provided a copyright management system based on digital watermarking and blockchain. They use the blockchain-based Inter Planetary File System (IPFS) to store and distribute the watermarked picture securely. Although these approaches have achieved traceability of information leakage without TTP, they have not given a corresponding accountability mechanism and unable to resolve the customer right problem.

Summarily, although there are many schemes to track information leakage, they all have some shortcomings such as TTP dependence, arbitration lacking and large overhead. In contrast, our scheme uses blockchain technology to achieve decentralized information leakage tracing and arbitration, which effectively solves the problem of the information leakage in a distributed environment.

VIII. Conclusion

By introducing the blockchain, homomorphic encryption public key system and LUT-based watermark embedding technology, we propose an information leakage tracking and arbitration mechanism. This mechanism enables service providers to fairly insert client watermarks into content and distribute content with the assistance of blockchain. This enables service providers to trace information leakers based on watermarks and use smart contracts for arbitration without the need for TTP participation.

As future work we consider constructing a blockchain platform dedicated to this solution, which presets the homomorphic encryption algorithm and the corresponding data structure, thereby further reducing the computational overhead and storage overhead of the blockchain system. We also plan to use some privacy protection technologies such as zero-knowledge proof and ring signature to protect the privacy of both parties in the process of transactions. In addition, how to ensure that the digital content provided by the service provider matches its published content description is also a problem that needs to be solved in the future.

References

[1] Y. Zhang, K. Guo, J. Ren, Y. Zhou, J. Wang, and J. Chen, "Transparent computing: A promising network computing paradigm," *Comput. Sci. Eng.*, vol. 19, no. 1, pp. 7–20, 2017.

[2] Y. Xu, J. Ren, Y. Zhang, C. Zhang, B. Shen, and Y. Zhang, "Blockchain empowered arbitrable data auditing scheme for network storage as a service," *IEEE Trans. Services. Comput.*,vol. 13, no. 2, pp. 289–300, Mar.–Apr. 2020.

[3] Y. Xu, J. Ren, G. Wang, C. Zhang, J. Yang, and Y. Zhang, "A blockchain-based nonrepudiation network computing service scheme for industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3632–3641, Jun. 2019.

[4] A. Gupta and R. K. Jha, "A survey of 5G network: Architecture and emerging technologies," *IEEE Access*, vol. 3, pp. 1206–1232, 2015.

[5] X. Cheng, Y. Wu, G. Min, and A. Y. Zomaya, "Network function virtualization in dynamic networks: A stochastic perspective," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 10, pp. 2218–2232, Oct. 2018.

[6] Gmobile Suppliers Association (GSA), "5G network slicing for vertical industries," 2017. [Online]. Available: https://gsacom.com/paper/5g-network-slicing-vertical-industries/

[7] B. Yin, Y. Wu, T. Hu, J. Dong, and Z. Jiang, "An efficient collaboration and incentive mechanism for internet-of-vehicles (IoVs) with secured information exchange based on blockchains," *IEEE Internet Things J.*, vol. 7, no. 3, pp. 1582–1593, Mar. 2020.

[8] L. Cheng *et al.*, "SCTSC: A semicentralized traffic signal control mode with attribute-based blockchain in IoVs," *IEEE Trans. Comput. Social Syst.*, vol. 6, no. 6, pp. 1373–1385, Dec. 2019., doi: 10.1109/TCSS.2019.2904633.

[9] S. Latif, J. Qadir, S. Farooq, and M. Imran, "How 5G wireless (and concomitant technologies) will revolutionize healthcare?," *Future Internet*, vol. 9, no. 4, pp. 93–117, 2017.

[10] I. Ahmad, T. Kumar, M. Liyanage, J. Okwuibe, M. Ylianttila, and A. Gurtov, "Overview of 5G security challenges and solutions," *IEEE Commun. Stand. Mag.*, vol. 2, no. 1, pp. 36–43, Mar. 2018.

[11] W. Sun, Z. Cai, Y. Li, F. Liu, S. Fang, and G. Wang, "Security and privacy in the medical Internet of Things: A review," *Secur. Commun. Netw.*, vol. 2018, pp. 1–9, 2018, Art. no. 5978636.

[12] Y. Xu, G. Wang, J. Ren, and Y. Zhang, "An adaptive and configurable protection framework against android privilege escalation threats," *Future Gener. Comput. Syst.*, vol. 92, pp. 210–224, 2019.

[13] N. Memon and P. W. Wong, "Protecting digital media content," *Commun. ACM*, vol. 41, no. 7, pp. 35–43, Jul. 1998.

[14] A. Khan, F. Jabeen, F. Naz, S. Suhail, M. Ahmed, and S. Nawaz, "Buyer seller watermarking protocols issues and challenges–a survey," *J. Netw. Comput. Appl.*, vol. 75, pp. 317–334, 2016.

[15] N. Memon and P. W. Wong, "A buyer-seller watermarking protocol," *IEEE Trans. Image Process.*, vol. 10, no. 4, pp. 643–649, Apr. 2001.

[16] C. Lei, P. Yu, P. Tsai, and M. Chan, "An efficient and anonymous buyer-seller watermarking protocol," *IEEE Trans. Image Process.*, vol. 13, no. 12, pp. 1618–1626, Dec. 2004.

[17] D. Bouslimi and G. Coatrieux, "A crypto-watermarking system for ensuring reliability control and traceability of medical images," *Signal Process. : Image Commun.*, vol. 47, pp. 160–169, 2016.

[18] Y. Hu, "A watermarking protocol for piracy tracing," in *Proc. Int. Symp. Electron. Commerce Secur.*, 2008, pp. 882–885.

[19] D. Hu and Q. Li, "A secure and practical buyer-seller watermarking protocol," in *Proc. Int. Conf. Multimedia Inf. Netw. Secur.*, 2009, pp. 105–108.

[20] S. Katzenbeisser, A. Lemma, M. U. Celik, M. van der Veen, and M. Maas, "A buyer–seller watermarking protocol based on secure embedding," *IEEE Trans. Inf. Forensics Secur.*, vol. 3, no. 4, pp. 783–786, Dec. 2008.

[21] Y. Peng, Y. Hsieh, C. Hsueh, and J. Wu, "Cloud-based buyer-seller watermarking protocols," in *Proc. IEEE SmartWorld, Ubiquitous Intell. Comput., Adv. Trusted Comput., Scalable Comput. Commun., Cloud Big Data Comput., Internet People Smart City Innov.*, 2017, pp. 1–9.

[22] M. Deng, T. Bianchi, A. Piva, and B. Preneel, "An efficient buyer-seller watermarking protocol based on composite signal representation," in *Proc. 11th ACM Workshop Multimedia Secur.*, 2009, pp. 9–18.

[23] C. Chen, C. Chen, D. Li, and P. Chen, "A verifiable and secret buyer–seller watermarking protocol," *IETE Tech. Rev.*, vol. 32, no. 2, pp. 104–113, 2015.

[24] T. Bianchi and A. Piva, "TTP-free asymmetric fingerprinting protocol based on client side embedding," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2014, pp. 3987–3991.

[25] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," pp. 1–9, 2008. [Online]. Available: https://bitcoin.org/bitcoin.pdf

[26] Z. Wang, Y. Tian, and J. Zhu, "Data sharing and tracing scheme based on blockchain," in *Proc. 8th Int. Conf. Logistics, Informat. Service Sci.*, 2018, pp. 1–6.

[27] R. Neisse, G. Steri, and I. Nai-Fovino, "A blockchain-based approach for data accountability and provenance tracking," in *Proc. 12th Int. Conf. Availability, Rel. Secur.*, 2017, pp. 1–10.

[28] Z. Meng, T. Morizumi, S. Miyata, and H. Kinoshita, "Design scheme of copyright management system based on digital watermarking and blockchain," in *Proc. IEEE 42nd Annu. Comput. Softw. Appl. Conf.*, 2018, vol. 2, pp. 359–364.

[29] D. Bhowmik and T. Feng, "The multimedia blockchain: A distributed and tamper-proof media transaction framework," in *Proc. 22nd Int. Conf. Digit. Signal Process.*, 2017, pp. 1–5.

[30] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *Proc. Int. Conf. Theory Appl. Cryptographic Techn.*, 1999, pp. 223–238.

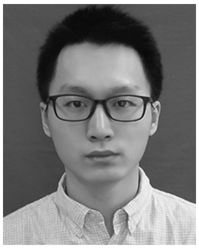
[31] M. Celik, A. Lemma, S. Katzenbeisser, and M. van der Veen, "Lookup-table-based secure client-side embedding for spread-spectrum watermarks," *IEEE Trans. Inf. Forensics Secur.*, vol. 3, no. 3, pp. 475–487, Sep. 2008.

[32] Ethereum, 2018. [Online]. Available: https://www.ethereum.org/

[33] M. B. Mollah, M. A. K. Azad, and A. Vasilakos, "Security and privacy challenges in mobile cloud computing: Survey and way ahead," *J. Netw. Comput. Appl.*, vol. 84, pp. 38–54, 2017.

[34] Q. Jia, L. Guo, Y. Fang, and G. Wang, "Efficient privacy-preserving machine learning in hierarchical distributed system," *IEEE Trans. Netw. Sci. Eng.*, vol. 6, no. 4, pp. 599–612, Oct.–Dec. 2019.

Yang Xu (Member, IEEE) received the Ph.D. degree from Central South University, Changsha, China, in 2019. From 2015 to 2017, he was a Visiting Ph.D. Student with Texas A&M University, USA. He is currently an Assistant Professor with the College of Computer Science and Electronic Engineering, Hunan University, Changsha, China. He has authored more than 30 articles in prestigious international journals and conferences, including IEEE TSC, TII, TCBB, FGCS, etc. He is working in the areas of network computing, blockchain, and operating system. He was the awardee of the Best Paper Award of IEEE IoP 2018. He is a member of IEEE, and a member of CCF Technical Committee on Blockchain. He served as a Steering Committee Chair and Program Committee Chair for IWCSS 2020, a Track Chair for IEEE CyberSciTech 2020 and CPSCom 2020, the Publicity Chair for ISSR 2019 and Ubisafe 2019, and as a Reviewer of more than 10 international journals.



Cheng Zhang (Student Member, IEEE) received the B.Sc. degree from the Shenyang University of Technology, Shenyang, China. He is currently working toward the master’s degree with the School of Computer Science and Engineering, Central South University, Changsha, China. His research interests mainly focus on the network security and blockchain.

Quanrun Zeng received the B.Sc. degree in information and computing science from Hunan Normal University, Changsha, China. He is currently working toward the master’s degree with the School of Computer Science and Engineering, Central South University, Changsha, China. His research interests mainly focus on the algorithms and access control.

Guojun Wang (Member, IEEE) received the B.Sc. degree in geophysics, the M.Sc. and Ph.D. degrees in computer science from Central South University, Changsha, China. He is a Pearl River Scholar Professor of Higher Education in Guangdong Province, a Doctoral Supervisor of the School of Computer Science, Guangzhou University. He had been a Professor with Central South University; an Adjunct Professor with Temple University, USA; a Visiting Scholar with Florida Atlantic University, USA; etc. He is a member of ACM and IEICE. His research interests include cloud computing, mobile computing, trustworthy/dependable computing, and cyber security.

Ju Ren (Member, IEEE) received the B.Sc., M.Sc., Ph.D. degrees in computer science from Central South University, Changsha, China in 2009, 2012, and 2016, respectively. From 2013 to 2015, he was a Visiting Ph.D. Student with the Department of Electrical and Computer Engineering, University of Waterloo, Canada. He is currently a Professor with the School of Computer Science and Engineering, Central South University, China, and an Associate Professor with the Department of Computer Science and Technology, Tsinghua University, China. He also serves as an Associate Editor for IEEE TVT and PPNA. His research interests include IoT, wireless communication, big data, artificial intelligence, and network computing.

Yaoxue Zhang (Senior Member, IEEE) received the B.Sc. degree from the Northwest Institute of Telecommunication Engineering, Xian, China, and the Ph.D. degree in computer networking from Tohoku University, Sendai, Japan, in 1982 and 1989, respectively. He is currently a Professor with the School of Computer Science and Engineering, Central South University, Changsha, China, and a Professor with the Department of Computer Science and Technology, Tsinghua University, Beijing, China. He has authored more than 200 papers. His research interests include computer networking, operating systems, and transparent computing. He is an Editor-in-Chief of *Chinese Journal of Electronics* and a Fellow of the Chinese Academy of Engineering.