# Revocable and Efficient Blockchain-based Fine-grained Access Control against EDoS Attacks in Cloud Storage

Qingyang Zhang, Chang Xu, Hong Zhong, Chengjie Gu, Jie Cui

***Abstract*—Users have become accustomed to storing data on the cloud using ciphertext policy attribute-based encryption (CP-ABE) for fine-grained access control. However, this encryption method does not consider the ability of malicious users to launch thousands of file download requests when launching an economic denial of sustainability attack (EDoS), which may be more expensive for data owners. Existing solutions typically use a cloud server to verify the download permissions of the data users. However, cloud servers are not completely trusted and cloud server providers and colluding data users can still launch an EDoS attack. With our scheme, using CP-ABE, a blockchain is introduced for verifying the download permission of data users. In addition, we propose a new mechanism to solve the problem of malicious user revocations under EDoS attacks by updating the ciphertext and symmetric encryption technology. A formal security proof has demonstrated that the proposed scheme is suitable for plaintext attack security. Theoretical and experimental analyses show that our scheme performs more efficiently than previous methods.**

***Index Terms*—access control, ciphertext-policy attributed-based encryption, cloud storage service, EDoS attacks, blockchain.**

## I. Introduction

In recent years, the rapid development of cloud storage services has attracted widespread attention from academia and industry owing to certain advantages such as always being turned on, a low cost, and flexible access [1]. Users can outsource data, including personal and business documents, to cloud storage and share them [2]. However, the data security issues that arise are not to be overlooked. Finding a way to ensure data security while maintaining the convenience of data sharing has become an urgent problem for cloud storage services. Attribute-based encryption (ABE) offers an effective solution to this challenge [3]. ABE allows users to combine access policies with data encryption. Users can achieve fine-grained access control over data without compromising the content of the data [4].

Although ABE offers a robust mechanism for protecting data confidentiality and enforcing fine-grained access control, there are still cross-layer attacks (outside the ABE layer), such as insider leakage attacks and Distributed Denial of Service (DDoS) attacks [5]. Insider leakage attacks involve employees within cloud service providers who may inadvertently, maliciously, or under duress leak sensitive data. On the other hand, DDoS attacks attempt to disrupt service availability by overwhelming the system with a flood of legitimate or fraudulent requests, preventing legitimate users' requests from being processed. Notably, a variant of DDoS attacks known as Economic Denial of Service (EDoS) has recently started to draw widespread attention [6]. Unlike traditional DDoS attacks, EDoS attacks aim to disrupt services and demand substantial additional payments, leading to significant financial losses [7]. This poses a serious threat to both businesses and individual users. For example, when a company rents a third-party cloud service for employees, the third-party cloud server inflates the number of file downloads but does not generate actual network traffic. The company will pay for the increased consumption resulting from such attacks, which can be a considerable and unreasonable financial burden.

Existing solutions typically require cloud servers to authenticate data users before downloading, meaning that the cloud server is considered a trusted entity and is expected to perform the permission validation without issues [7]. However, this assumption is problematic in real-world applications because cloud servers are not entirely trustworthy. There are mainly two reasons for this. Firstly, cloud servers might be curious and attempt to peek at the data out of sheer curiosity. Secondly, and more importantly, due to the presence of multi-tenancy, a user on the cloud might break through the management restrictions and spy on other tenants' data. Therefore, when considering data access control in cloud environments, we need to re-examine the impact of this semi-trustworthiness. Recently, some solutions have suggested that data users generate a "download request" and send it to the cloud server, which then forwards the request to a central authority for validation [8]. Although this method avoids the cloud server directly processing users' download requests, it does incur some overhead for the central authority. Usually, central authority should not perform such authentication operations and should only be with key management. Additionally, the cloud server can still obtain some private information by observing the data user's "download request." Therefore, the current solutions are not yet perfect, and further research and improvement are needed.

As we are aforementioned, current works have limitations on the workload of CA. Recently, blockchain technology, as an innovative technology that has emerged in recent years, has

Q. Zhang, C. Xu, H. Zhong and J. Cui are with the Key Laboratory of Intelligent Computing and Signal Processing of Ministry of Education, School of Computer Science and Technology, Anhui University, Hefei 230039, China, and the Anhui Engineering Laboratory of IoT Security Technologies, Anhui University, Hefei 230039, China (e-mail: zhongh@ahu.edu.cn).

C. Gu is with the School of Public Security and Emergency Management, Anhui University of Science and Technology, Hefei, 231131, China, and the Security Research Institute, New H3C Group, Hefei, 230088, China (e-mail: gcj@ustc.edu.cn).

attracted significant attention due to its unique characteristics [9]. Firstly, blockchain employs a distributed ledger technology, where all transaction records are publicly transparent. This transparency is a key factor contributing to the high trust in blockchain systems [10]. Secondly, the immutability of blockchain provides a high degree of security for data. Lastly, the decentralized nature of blockchain also enhances the system's credibility. In a blockchain network, data is distributed across multiple nodes in the network [11]. This decentralized structure reduces the risk of single-point failures and improves the overall stability and reliability of the system. Therefore, we consider using blockchain technology to verify data users before downloading, thereby assisting in resolving the EDoS attack problem.

In addition to the issue of preventing EDoS attacks due to the semi-trustworthiness of cloud servers, we should also pay attention to the problem of attribute/permission revocation in the context of EDoS attacks. Current revocation mechanisms are inefficient [12], especially in access control systems where multiple users share each attribute. Revoking any single attribute or user affects other users, introducing significant computational overhead and potentially leading to a significant decline in system performance [13]. To address this issue, we aim to build an efficient management of user attributes based on a binary tree structure, aka Key Encryption Key (KEK) tree, which allows rapid retrieval to minimize the impact of revocation operations on the system. Specifically, we will allocate a unique path key to each user based on their attributes. When revoking an attribute for a specific user, the system can quickly locate the corresponding key and update only the parts related to that user, thus avoiding the widespread impact inherent in traditional revocation mechanisms.

### A. Our contribution

In response to the above problems, we design a scheme to achieve fine-grained access control while defending EDoS attacks. The following are our contributions:

- We design a fine-grained access control scheme that can defend against an EDoS attack. In our scheme, the hash of the challenge plaintext and the corresponding challenge ciphertext generated by the data owner are uploaded to the blockchain. The blockchain then verifies whether the data users have permission to access the data, which avoids the problem of semi-trusted cloud providers and malicious users colluding to launch an EDoS attack.
- The proposed scheme improves the performance of revocation for malicious users by utilizing key encryption key trees to update the challenge ciphertexts. In addition, we use two-layer symmetric encryption technology to improve the CP-ABE system, avoiding the problem of revoked users still being able to decrypt the updated challenge ciphertext.
- Based on the decision linear assumption (DLIN) assumption, our scheme is secure under chosen plaintext attacks. Theoretical and experimental analysis show that our scheme is more efficient than previous approaches.

### B. Structure of the Paper

Section II introduces the related work of this paper. The necessary preliminaries discussed in this paper will be covered in Section III. The system model, which consists of the system architecture and the security model, is then presented in Section IV. In Section V, we will next introduce a certain structure. In Sections VI and VII, respectively, we give security analysis and performance analysis. Section VIII provides this paper's conclusion.

## II. Related Work

### A. Defend against EDoS attacks.

Although CP-ABE can support fine-grained data access, it cannot protect data from many other types of attacks. For example, malicious insider attack, denial of service (DoS) Attack, and so on. Because in a cloud environment, internal personnel with access permissions may exploit their privileges to attack the data. Additionally, if access permissions are not properly set, unauthorized users may gain access to shared data, leading to data breaches or malicious tampering. Several countermeasures against attacks have been proposed in the work [14], [15] and [16]. But Xue et al. [7] indicated that earlier works had not been fully protected against EDoS attacks at the algorithmic level. They further proposed a new scheme in response to this attack. They check whether the user is authorized to prevent EDoS attack. They pointed out that cloud providers verify that the data user has permission to download the file by generating some random challenge plaintext and corresponding ciphertext related to the access policy. However, this verification method inevitably increases the cloud's network bandwidth cost, which may not be available to some service users with pay-as-you-go plans. Then, Xue et al. proposed a resource accounting protocol to prevent semi-trusted cloud providers from cheating data owners. Evidence of downloaded files is recorded through a resource accounting agreement. Although the proposed resource consumption protocol guarantees the cloud provider's transparency to the data owner, it also increases the computational overhead of the data user. Ning et al. [8] also proposed a corresponding solution for EDoS attacks. They proposed that data users can generate "download requests" and send them to the cloud server for further to central authority, which will ultimately verify the data user's permissions. However, this approach inevitably adds computing and communication overhead to the central authority.

### B. Blockchain based ABE.

Recent results have shown that combining blockchain with cloud has become popular. Ding et al. [17] proposed an attribute-based access control scheme for IoT systems that simplifies access management by defining a set of attributes for each device. By utilizing blockchain technology, the scheme ensures the immutability and traceability of these attributes, thereby enhancing the security and efficiency of IoT systems. Ourad et al. [18] proposed a blockchain-based solution for achieving authentication and secure communication with IoT

devices. It leveraged the intrinsic features of blockchain to enhance the system's accountability, integrity, and tamper-proof logs. Additionally, they detailed the overall system design and architecture and the testing and implementation of a realistic scenario as a proof of concept. Yu et al. [19] introduced a blockchain-enhanced security access control scheme for IIoT in smart factories, ensuring secure storage, access control, and malicious user tracking through a unified identity authentication process and domain-specific security policies. Wan et al. [20] introduced a blockchain architecture to reshape the traditional Industrial Internet of Things (IIoT) architecture, addressing the security and privacy concerns arising from the increasing number of nodes in large-scale networks. To address the vulnerabilities and inefficiencies of existing Industrial Internet of Things (IIoT) systems, [21] proposed a novel system that combines blockchain with the Internet of Things (IoT). This system employs a credit-based consensus mechanism and introduces a new credit-based Proof of Work (PoW) mechanism to ensure the security and transaction efficiency of the system. Lee et al. [22] came up with using blockchain to provide a trustworthy and secure data collection environment by merging deep machine learning with Ethereum.

### C. Revocation in ABE.

To address the issue of malicious data user revocation in cloud environments, [23] proposed an ABE scheme that supported proxy re-encryption. The scheme partially hides access structures, preventing receivers from extracting sensitive information from the ciphertext. Additionally, the security of the scheme can be reduced to the Decisional Bilinear Diffie-Hellman (DBDH) assumption and the Decisional Linear (DL) assumption. [24] proposed an access control scenario that supported direct revocations. During the encryption phase, the revocation list is also encrypted in the ciphertext. However, this scenario cannot address the issue of updating access policies, and [25] also had a similar problem. A revocable attribute-based encryption approach with integrity verification was put out by Ge et al. [26]. They introduced a new security requirement—data integrity protection for revocable attribute-based encryption (RABE). However, this scheme is relatively inefficient because it adds an access policy at the time of revocation. Hur et al. [27] further proposed an attribute-based encryption scheme with efficient user and attribute revocation. In their scenario, they enabled access control for users, which enhanced backward/forward secrecy for any member of the attribute group.

Although the above studies have successfully achieved fine-grained access control, these schemes mainly focus on the encryption and hiding of access structures without fully considering how to resist EDoS attacks. Furthermore, existing revocation mechanisms often do not handle updating access policies well. This is a significant flaw when responding to EDoS attacks, as attackers may exploit the inflexibility of system policies to launch their attacks. Therefore, solving the above problems in the cloud environment remains a challenge.

TABLE I
NOTATIONS

| Notations | Definitions |
|---|---|
| $\varphi$ | Security parameter |
| $\rho$ | Access policy |
| $N_t$ | The $t$-th row of the matrix $N$ |
| $N_{t,j}$ | The $(t,j)$-th element of the matrix $N$ |
| $J$ | Attribute group |
| $J_S$ | an attribute group related to ciphertexts |
| $J_s$ | a certain attribute in the $J_S$ |
| $PK_i$ | The path key of the data user on the KEK tree |
| $v_i$ | Node $i$ of the KEK tree |
| $KEK_i$ | The key on the node $i$ |
| $negl(\varphi)$ | Negligible function of $\varphi$ |
| $H$ | Secure hash function that map $\{0,1\}* \rightarrow Z_p$ |
| $pk, msk$ | System of public key and master secret key |

## III. PRELIMINARIES

In this section, we will describe the preliminaries associated with our scheme. All the notations used in this paper are summarized in Table I.

### A. Access Structure and Linear Secret Sharing Schemes

**Definition 1.** *(Access structure [28]).* Let $\{B_1, \ldots, B_n\}$ be a set of attributes. A collection $\mathbb{B} \subseteq 2^{\{B_1,\ldots,B_n\}}$ is monotone if $\forall C, D$: when $C \in \mathbb{B}$ and $C \subseteq D$, then $D \subseteq \mathbb{B}$.

A monotone access structure is a monotone collection $\mathbb{B}$ of non-empty subsets of $\{B_1, \ldots, B_n\}$, i.e. $\mathbb{B} \subseteq 2^{P_1,\ldots,P_n} \setminus \{\emptyset\}$. The sets in $\mathbb{B}$ are permitted sets, whereas the sets outside of $\mathbb{B}$ are prohibited sets.

**Definition 2.** *(Linear secret sharing schemes).* Let $T = \{t | t \in \{1, \ldots, m_1\}, \rho(t) \in S\}$ be the rowset in $N$ that belongs to $S$, and let $S$ be a set of characteristics. If there is a linear combination of rows in $T$ that results in $(1, 0, \ldots, 0)$, then we say that $(N, \rho)$ accepts $S$. More formally, there should be coefficients $\{\gamma_t\}_{t\in T}$ such that

$$\sum_{t\in T} \gamma_t (N)_t = (1, 0, \ldots, 0)$$

where $(N)_t$ is the $t$th row of $N$. When Lewko and Water's methods are applied to a Boolean formula, then a coefficient of 0 or 1 can always be chosen for the resulting LSSS, regardless of the set $S$.

### B. Bilinear Map and Decision Linear Assumption

Let $G$, $H$ and $G_T$ be three cyclic multiplicative groups of prime order $p$. Let $g$ be the generators of $G$ and $h$ be the generator of $H$. The bilinear pairing is a map $e : G \times H \rightarrow G_T$ with the following features:

- **Bilinear:** For all $g \in G$, $h \in H$ and $c, d \in Z_p$, we can achieve that $e(g^c, h^d) = e(g, h)^{cd}$
- **Non-degenerate:** $e(g, h) \neq 1$.
- **Computability:** It is high efficiency to compute $e(g, h)$ for all $g \in G$ and $h \in H$.

The superiority for all probabilistic polynomial time (PPT) adversaries $E$ in settling the decision linear problem is defined as

$$Edv^E_{DLIN}(\varphi) := |Pr[E(1^\varphi, par, D, T_0) = 1] - Pr[E(1^\varphi, par, D, T_1, ) = 1]| \leq negl(\varphi)$$

where $par := (p, G, H, G_T, e, g, h)$; $x_1, x_2 \leftarrow Z_p^*$, $r_1, r_2, r \leftarrow Z_p$; $D := (g^{x_1}, g^{x_2}, h^{x_1}, h^{x_2}, g^{x_1 r_1})$, $T_0 := (g^{r_1+r_2}, h^{r_1+r_2})$; $T_1 := (g^r, h^r)$.

The probability is taken over the randomness used by $E$. Keep in mind that it is difficult for all probabilistic polynomial time adversaries $E$ to tell $T_0$ from $T_1$ for $par \leftarrow Gen, x_1, x_2, r_1, r_2, r \leftarrow Z_p$.

### C. Blockchain and Smart Contract

Blockchain technology [9] is a decentralized ledger technology characterized by its decentralized nature, transparency, immutability, and traceability. In the field of data sharing, blockchain technology can enhance the security and transparency of data. By leveraging blockchain technology, a secure authentication mechanism can be established where users need to pass blockchain verification before downloading files, which can effectively prevent and mitigate EDoS attacks.

Smart contracts [29] are computer protocols that automatically execute, control, or document legally binding events and actions. They are stored on the blockchain and are an important component of blockchain technology. The execution results of smart contracts are recorded on the blockchain, thereby providing transparency and traceability. Each time a user accesses or downloads a file, it is recorded on the blockchain, ensuring the transparency and traceability of the operation.

### D. KEK Tree

A binary tree structure was introduced to implement efficient revocation. This binary tree is called a KEK (key encryption key) tree, and the nodes of this KEK tree are KEK keys. Here, we briefly introduce the revocation technique. Let $U$ be the collection of users in the system and $R$ to be the revocation list [30]. The binary tree is expressed as:

- The tree's leaf nodes represent each member of $U$. Each internal node is assigned a randomly generated key. Each member $u_i \in U$ gains the path keys $PK_i$ from its leaf node to the root node of the tree securely. For instance, $u_2$ stores $PK_2 = KEK_9, KEK_4, KEK_2, KEK_1$ as its path keys in Fig. 1.
- $J$ $J_S$, $J_s$ and $v_i$ are defined as an attribute group, an attribute group related to ciphertexts, a certain attribute in the $J_s$ and a node, respectively. We define a minimum set of nodes, which can override all the leaf nodes associated with users in $J_s$. For example, in Fig. 1, for $U' = u_1, u_2, u_3, u_4, u_5, u_6$ in $J_s$, $v_2$ and $v_6$ are the root nodes of the minimum cover sets that can override all of the members in $J_s$. The set of node keys in the minimum cover set is defined as $KEK(J_s, U')$, so in this example, $KEK(J_s, U') = \{KEK_2, KEK_6\}$. It follows that this collection overrides these users in $J_s$ and only them. Any user who does not have the $J_s$ attribute cannot know the key of the KEK tree under this attribute.

## IV. System Model

This section describes the system framework and security model in detail.

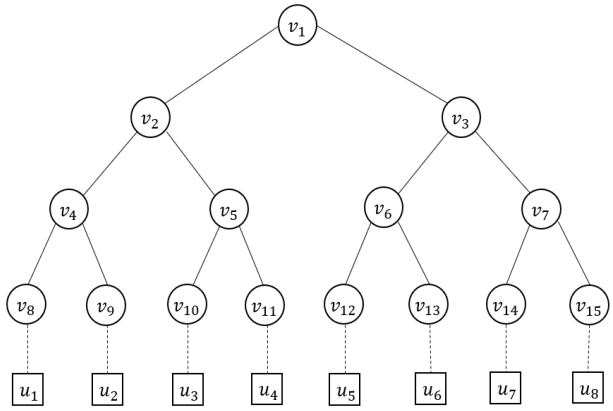



Fig. 1. An example of a KEK tree with eight users.

### A. System Architecture

In this subsection, we will introduce our system model and briefly describe the system workflow, followed by the definition of the six algorithms used in our scheme.

As shown in Fig. 2, the cloud storage system consists of five entities: cloud server, blockchain, central authority, data owner, and data user.

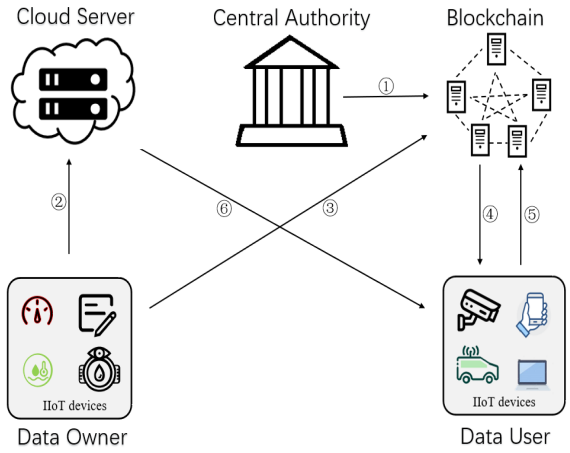



Fig. 2. System.

- **Cloud Server (CS)**: The cloud server is semi-trusted party. It offers data owners a service for data storage. The cloud server has enough processing and storage capacity to hold the data and reply to the blockchain's request.
- **Blockchain (BC)**: A blockchain is fully trusted party. It authenticates the data user before downloading the file.
- **Central Authority (CA)**: The system is initially configured by CA, which also issues keys for the user (the data owner and the data user).
- **Data Owner (DO)**: A data owner is the publisher of the file, which is uploaded to the cloud in encrypted form. In our system, the data owner is not always online.

- **Data User (DU)**: The data user is a role that wants to gain certain cloud-based files. Before downloading files, they must be certified by the blockchain (to prevent EDoS attacks).

Our scheme consists of six algorithms, i.e., Setup, KeyGen, Encrypt, Challenge Generation, Decrypt, and CTUpdate. With the above six algorithms, our system is working as follows.

① The CA runs the Setup and KeyGen algorithms to generate the public and private keys required by the system. The blockchain receives the system's public key and a set of attributes from the central authority. It is to help blockchain verify data users' download rights.

② Utilizing the Encrypt algorithm, the data owner encrypts the data file with attribute-based encryption and KEK tree and uploads it to the cloud server. The cloud server provides storage data services.

③ When the data owner encrypts the ciphertext, in addition to the data ciphertext, some challenge ciphertexts are also generated using the Challenge Generation algorithm. The data owner uploads these challenge ciphertexts to the blockchain.

④ When a data user requests to download a file, the blockchain first verifies the data user's permission to download. The blockchain sends challenge ciphertexts $enchai_i$ to the data user. The blockchain verifies the download permission of the data user through the $chal_i$ and $enchal_i$, obtained by the Decrypt algorithm.

⑤ The data user sends challenge the hash of decryption $hash_i'$ to the blockchain, the blockchain verifies that $hash_i'$ is correct.

⑥ If the blockchain is validated correctly, the data user receives the ciphertexts from the cloud server.

Here, we give the algorithms included in the system:

- **Setup**$(1^{\varphi}) \rightarrow (msk, mpk)$: When entering security parameters, the algorithm gives back the master key $msk$ and the master public key $mpk$.
- **KeyGen**$(msk, S, u_i) \rightarrow (sk, KEK_i)$: When entering the master key $msk$, a group of attributes $S$ and identity $u_i$, the algorithm returns key $sk$ and $KEK_i$.
- **Encrypt**$(mpk, (N, \rho), sk, k', KEK_i) \rightarrow CT$: When entering the master public key $mpk$, access structure $(N, \rho)$, the symmetric key $k'$, the data owner's key $sk$ and the key $KEK_i$, the algorithm returns a ciphertext $CT$.
- **Challenge Generation**$(chal_I, k') \rightarrow (c_1, c_2)$: When entering the challenge plaintexts $chal_I$ and symmetric key $k'$, the algorithm returns ciphertext $c_1$ and $c_2$.
- **Decrypt**$(CT, sk, KEK_i) \rightarrow k'$: When entering the ciphertext $CT$, data user's key $sk$ and $KEK_i$, the algorithm returns symmetric key $k'$ or an valid symbol $\perp$.
- **CTUpdate**$(CT, p_1', p_2', KEK_i') \rightarrow CT'$: When entering the ciphertext $CT$, random numbers $p_1', p_2'$ and key $KEK_i'$, the algorithm returns updated ciphertext $CT'$.

### B. Security Assumptions and Design Objectives

The security assumptions for each entity are described below.

- Cloud server is a semi-trusted party in the sense that it may charge more to the data owner. Specifically, it can honestly store data for the data owner. However, it may exploit vulnerabilities to exaggerate the resource consumption of data owners and collude with malicious data users to launch EDoS attacks.
- Blockchain is a fully trusted party in the sense. It will honestly verify the download permission of the data user.
- Central authority is fully trusted party by other entities.
- The Data owner is honest that she/he encrypts the data and uploads the encrypted data to the cloud and the blockchain.
- Data user is malicious because she/he may try to download an unauthorized shared file and launch an EDoS attack.

Taking into account the security assumptions of each of the above entities, the primary design objectives of our proposed system include:

- **Confidentiality of shared data.** Cloud servers and unauthorized data users cannot query any data uploaded to the cloud by the data owner.
- **Access control on download request.** To prevent EDoS attacks by malicious data users, only authorized data users can download the data.
- **Access control on shared data.** Only authorized data users can decrypt the data.

### C. Security Requirements

Taking into account the security assumptions of each of the above entities and design targets, the security requirements of our proposed system include:

- **Security against semi-trusted cloud server:** The cloud server has no access to data plaintexts stored on it by the data owner.
- **Security against malicious data user:** (a) Any data user who does not have permission to download cannot download shared files; (b) Any data user who does not have download permissions will also not be able to decrypt the file after it is gained. If a data user's attribute set fails to comply with the access policy for a shared file, the user is defined as having no permissions.
- **Security against malicious data user and cloud server collusion:** Data users and cloud servers cannot collude to exaggerate the resources consumed by data download and storage.

**Definition 3.** *An attribute based access control encryption scheme [4] is secure against all PPT adversaries $E$ if their*

$$Edv_{\prod}^{E}(\varphi) := |Pr[Expt_{\prod,E}(\varphi, 0) = 1] - Pr[Expt_{\prod,E}(\varphi, 1) = 1]|$$

*is negligible in $\varphi$.*

Setup: The challenger $C$ executes setup$(1^{\varphi})$ to gain $mpk, msk$, and provides $mpk$ to $E$.

Key query: Upon entering a set of attributes $S$, the challenger $C$ returns a secret key $sk = KeyGen(msk, S)$ and hands it to $E$.

Challenge: After entering a pair of $chal(chal_0, chal_1)$ and an access structure $(N, \rho)$, the challenger $C$ returns a ciphertext $CT = Encrypt(mpk, (N, \rho), chal_b)$ and hands it to $E$.

This security definition effectively guarantees fine-grained access control for data users. An attribute-based access control encryption scheme meets the security of this definition, which means that unauthorized data users cannot decrypt challenge ciphertext and data files.

## V. Main Construction

### A. Overview of Our Construction

Based on the work of FAME [4] and Xue et al. [7], we have proposed an access control scheme that can prevent EDoS attacks. As previously mentioned, our scheme consists of six algorithms: Setup, KeyGen, Encrypt, Challenge Generation, Decrypt, and CTUpdate.

In our proposed solution, we leverage blockchain technology to authenticate data users' download privileges. Specifically, the data owner generates challenge plaintexts and their corresponding ciphertexts, which are then jointly uploaded to the blockchain. When a data user requests to download a file, the blockchain distributes the challenge ciphertext to the user. Access is granted if the user successfully decrypts the ciphertext and denies otherwise. Then, we employ KEK trees and two-layer symmetric encryption to revoke malicious users. Specifically, CA constructs binary trees based on users' attributes, assigning a key to each node, which is used to encrypt the ciphertext. When revoking attributes of a specific user, the system can quickly locate the corresponding keys and only update the parts related to that user, thus minimizing the impact of the revocation operation on the system. The two-layer symmetric encryption mechanism can avoid the problem that users can still decrypt the updated challenge ciphertext after being revoked.

### B. Setup

CA runs an asymmetric group generator $Gen(1^\varphi)$ to gain $(p, G, H, G_T, e, x, y)$. Then, it picks $b_1$, $b_2 \leftarrow Z_p^*$ and $a_1$, $a_2$, $a_3 \leftarrow Z_p$, and computes $H_1 := y^{b_1}, H_2 := y^{b_2}, T_1 := e(x, y)^{a_1 b_1 + a_3}$, $T_2 := e(x, y)^{a_2 b_2 + a_3}$. Finally, the public key can be set as $pk = (y, H_1, H_2, T_1, T_2)$ and the CA could pick $c_1$, $c_2 \leftarrow Z_p^*$ and let the master secret key $msk = (x, y, b_1, b_2, c_1, c_2, x^{a_1}, x^{a_2}, x^{a_3})$.

### C. KeyGen

The key generation phase is executed by CA and is divided into the following two sub-phases: the attribute key generation and the KEK generation.

**Attribute Key Generation**: Picks $r_1, r_2 \leftarrow Z_p$, uses $h, c_1, c_2$ from $msk$ and computes $sk_0 := (y^{c_1 r_1}, y^{c_2 r_2}, y^{r_1 + r_2})$. Then, for all $s \in S$ and $z = 1, 2$, CA picks $\sigma_s \leftarrow Z_p$ and computes:

$$sk_{s,z} := H(s1z)^{\frac{c_1 r_1}{b_z}} \cdot H(s2z)^{\frac{c_2 r_2}{b_z}} \cdot H(s3z)^{\frac{r_1 + r_2}{b_z}} \cdot x^{\frac{\sigma_s}{b_z}}$$

and sets $sk_s := (sk_{s,1}, sk_{s,2}, g^{-\sigma_s})$. Also, picks $\sigma' \leftarrow Z_p$, computes:

$$sk'_z := g^{a_z} \cdot H(011z)^{\frac{c_1 r_1}{b_z}} \cdot H(012z)^{\frac{c_2 r_2}{b_z}} \cdot H(013z)^{\frac{r_1 + r_2}{b_z}} \cdot x^{\frac{\sigma'}{b_z}}$$

for $z$=1,2 and set $sk' = (sk'_1, sk'_2, x^{a_3} \cdot x^{-\sigma'})$ and let the secret key $sk$ is $(sk_0, \{sk_s\}_{s \in S}, sk')$.

**KEK Generation**: The CA runs $KEKGen(U)$ and generates KEK keys for users in $U$. First, the CA sets binary KEK trees for the universe of users $U$ as in Fig.1, which will be used to distribute the attribute group keys to users in $u \in U$. Each node $v_i$ in the tree has a key $KEK_i$. A set of $KEK_i$ on the path nodes from a leaf to the root are called path keys.

### D. Encrypt

This part of the operation is performed independently by the data owner. The message is encrypted by the data owner using hybrid encryption. The process is as follows:

First, the data owner uses a symmetric encryption algorithm [31], chooses a symmetric key $k \leftarrow \{0, 1\}^\varphi$ at random that is used to encrypt the message $msg$:

$$AEAD.Enc(msg, k) \rightarrow ct_k$$

Second, the data owner chooses a symmetric key $k' \leftarrow \{0, 1\}^\varphi$ at random that encrypts the symmetric key $k$:

$$AEAD.Enc(k, k') \rightarrow Enck$$

The data owner then uses public key $pk$ and access policy $\mathbb{A}$ to encrypt the symmetric key $k'$ with CP-ABE and the data owner picks $p_1, p_2 \leftarrow Z_p$ and computes:

$$ct_0 := (H_1^{p_1}, H_2^{p_2}, y^{p_1 + p_2})$$

Next, for any $J_s \in J_S$, the data owner uses the key $KEK_i$ and supposes that $N$ has $m_1$ rows and $m_2$ colums. Then for $t = 1, \ldots m_1$, and $n = 1, 2, 3$, it computes:

$$ct := T_1^{p_1} \cdot T_2^{p_2} \cdot k'$$
$$ct_{t,n} := [H(\rho(t)n1)^{p_1} \cdot H(\rho(t)n2)^{p_2} \cdot \prod_{j=1}^{m_2} [H(0jn1)^{p_1} \cdot H(0jn2)^{p_2}]^{(N)_{t,j}}]^{KEK_i}$$

and sets $ct_t := (ct_{t,1}, ct_{t,2}, ct_{t,3})$.

Then, the data owner selects root nodes of the minimum cover sets in the KEK tree that can cover all of the leaf nodes associated with users in $J_s$. This collection covers all users in $J_s$ and only them, and any user $u \notin J_s$ can by no means know any KEK in $KEK(J_s, U')$. Next, for any $J_s \in J_S$, $KEK_i \in KEK(J_s, U')$, the data owner chooses a symmetric key $K_i \leftarrow \{0, 1\}^\varphi$ at random to encrypt the $KEK_i$ and generates a header message:

$$Kdr = (\forall KEK_i \in KEK(J_s, U') : AEAD.Enc(KEK_i, K_i)),$$

This encryption is used as a means of distributing the attribute group keys to authorized users. It is crucial to understand that the method of distributing attribute group keys via $Kdr$

operates in a stateless manner. This ensures that, regardless of the practical challenges in constantly updating users' key states, users retain the ability to decrypt the attribute group key received through $Kdr$ at any given moment, on the condition that their access has not been revoked from the respective attribute groups and they possess the necessary authorization for decryption.

Finally, lets the ciphertext $CT = (ct_0, ct_{1,n}, \ldots, ct_{m_1,n}, ct, ct_k, Enck, Kdr)$.

### E. Challenge Generation

First, the data owner chooses $M$ random challenge plaintexts:

$$\{chal_1, chal_2, \ldots, chal_M\}, chal \leftarrow \{0,1\}^{\varphi}$$

and they should be different from each other.

Then, the data owner creates these challenges' hashes: $hash_I = H(chal_I), \forall I \in [1, M]$, where $H()$ is a collision-resistant hash function.

And the data owner uses $k'$ to encrypt each challenge plaintext $chal_I$ with the fixed suffix "challenge". The suffix is used to distinguish between challenges and data plaintext, which prevents the cloud from sending data to the blockchain as a challenge. The encryption is likewise protected by the same symmetric encryption in this case:

$$AEAD.Enc(chal_I, k') \rightarrow enchal_I$$

Now, we have $ch_1 = \{encha_I\}_{I\in[M]}, ch_2 = \{hash_I\}_{I\in[M]}$. And $(ch_1, ch_2)$ is uploaded to the blockchain.

### F. Decrypt

When a data user requests to download data, the blockchain will choose a new $enchal_I$ and send the $enchal_I$ to the data user. Specifically, the data user first decrypts $KEK_i$ using the symmetric encryption key $K_i$,

$$AEAD.Dec(K_i, Kdr) \rightarrow KEK_i$$

Second, recall that if the set of attributes $S$ in $sk$ satisfies the access structure $(N, \rho)$ in ciphertext , then there exists constants $\{\gamma_t\}_{t\subset T}$. Now, compute

$$num := ct \cdot e(\prod_{t\in T} ct_{t,1}^{\gamma_t \cdot \frac{1}{KEK_i}}, sk_{0,1}) \cdot e(\prod_{t\in T} ct_{t,2}^{\gamma_t \cdot \frac{1}{KEK_i}}, sk_{0,2})$$
$$\cdot e(\prod_{t\in T} ct_{t,3}^{\gamma_t \cdot \frac{1}{KEK_i}}, sk_{0,3})$$

$$den := e(sk_1' \cdot \prod_{t\in T} sk_{\rho(t),1}^{\gamma_t}, ct_{0,1}) \cdot e(sk_2' \cdot \prod_{t\in T} sk_{\rho(t),2}^{\gamma_t}, ct_{0,1})$$
$$\cdot e(sk_3' \cdot \prod_{t\in T} sk_{\rho(t),3}^{\gamma_t}, ct_{0,3})$$

and output $k' = \frac{num}{den}$.

Then, decrypt $chal_I'$,

$$AEAD.Dec(k', enchal_I) \rightarrow chal_I'$$

The blockchain receives $chal_I'$ from the data user.

Eventually, the data user decrypts $k$ using the symmetric encryption key $k'$, which can further be used to decrypt the $ct_k$.

$$AEAD.Dec(k', Enck) \rightarrow k$$

$$AEAD.Dec(k, ct_k) \rightarrow msg$$

### G. CTUpdate

When some users cause the data to be leaked, the members of the attribute group need to be changed. Specifically, the blockchain will send the information that needs to be changed to the CA and the data owner. When the CA receives the information of the member of the blockchain, it will change the key set of the attribute group affected by the members. In addition, the data owner will update the ciphertext.

First, the CA selects new minimum cover sets for $J_s'$, including a new joining user who comes to hold an attribute or excluding a leaving user who comes to drop an attribute. Then, for any $J_s' \in J_S$, $KEK_i' \in KEK(J_s', U'')$, chooses a new symmetric key $K_i' \leftarrow \{0,1\}^{\varphi}$ at random number is used to encrypt the $KEK_i'$ and generates a new header message:

$$Kdr' = (\forall KEK_i' \in KEK(J_s', U'') : AEAD.Enc(KEK_i', K_i')),$$

Next, the data owner randomly re-chooses a symmetric key $\{0,1\}^{\varphi} \leftarrow k''$ to encrypt $k$:

$$AEAD.Enc(k, k'') \rightarrow Enck'$$

Then, the data owner chooses $p_1', p_2' \in Z_p^*$, and a $KEK_i'$ to re-encrypt the ciphertext $ct$, $ct_{t,n}$ and $ct$. It computes:

$$ct_0' := (H_1^{p_1+p_1'}, H_2^{p_2+p_2'}, y^{p_1+p_1'+p_2+p_2'})$$
$$ct_{t,n}' := [H(\rho(t)n1)^{p_1+p_1'} \cdot H(\rho(t)n2)^{p_2+p_2'}$$
$$\cdot \prod_{j=1}^{m_2} [H(0jn1)^{p_1+p_1'} \cdot H(0jn2)^{p_2+p_2'}]^{(N)_{t,j}}]^{KEK_i'},$$
$$ct' := T_1^{p_1+p_1'} \cdot T_2^{p_2+p_2'} \cdot k''$$

and outputs the updated ciphertext $CT' = (ct_0', ct_{1,n}', \ldots, ct_{1,m_1}', ct', Enck', Kdr')$

The key update process assures fine-grained access control for data users, such as the instant revocation of a user in each attribute group. Additionally, user-level revocation can be done in each property instead of the system level. Therefore, though a data user is revoked in one attribute group, he can still access the data under other attribute groups that satisfy the access policy.

## VI. Security Analysis

### A. Correctness analysis

When $S$ satisfies $(N, \rho)$, we indicate that the correct message with probability one is recovered. For $n = 1, 2, 3$,

$$\begin{aligned}\prod_{t\in T} ct_{t,n}^{\gamma_t} &= \prod_{t\in T}(H(\rho(t)n1)^{\gamma_t p_1}) \cdot H(\rho(t)n2)^{\gamma_t p_2} \\ &\cdot \prod_{j=1}^{m_2}[H(0jn1)^{p_1} \cdot H(0jn2)^{p_2}]^{\gamma_t(N)_{t,j}} \\ &= (\prod_{j=1}^{m_2}[H(0jn1)^{p_1} \cdot H(0jn2)^{p_2}]^{\sum_{t\in T}\gamma_t(N)_{t,j}}]) \\ &\cdot (\prod_{t\in T} H(\rho(t)n1)^{\gamma_t p_1} \cdot H(\rho(t)n2)^{\gamma_t p_2}) \\ &= H(0jn1)^{p_1} \cdot H(0jn2)^{p_2} \cdot \prod_{t\in T} H(\rho(t)n1)^{\gamma_t p_1} \\ &\cdot H(\rho(t)n2)^{\gamma_t p_2}\end{aligned}$$

At present, the product of every term in num except the first is provided by

$$\begin{aligned}&\prod_{z\in\{1,2\}}[e(H(011z), y)^{b_1 r_1 p_z} \cdot e(H(012z), y)^{b_2 p_2 p_z} \\ &\cdot e(H(013z), h)^{(r_1+r_2)p_z} \cdot \prod_{t\in T}(e(H(\rho(t)2z)^{\gamma_t}, y)^{b_2 r_2 p_z} \\ &\cdot e(H(\rho(t)3z)^{\gamma_t}, y)^{(r_1+r_2)p_z}]\end{aligned}$$

When the above products are divided by dem, we can easily see that only the opposite is left (the rest of the terms are canceled)

$$\begin{aligned}&(\prod_{z\in\{1,2\}} e(x^{a_z} \cdot x^{\frac{\sigma'}{b_z}} \cdot \prod_{t\in T} g^{\frac{\gamma_t \sigma_{\rho(t)}}{b_z}}, y^{b_z p_z}) \\ &\cdot e(x^{a_3} \cdot x^{-\sigma'} \cdot \prod_{t\in T} x^{-\gamma_t \sigma_{\rho(t)}}, y^{p_1+p_2})))\end{aligned}$$

which is just equal to $e(x, y)^{a_1 b_1 p_1 + a_2 b_2 p_2 + a_3(p_1+p_2)}$, Hence, $k'$ is successfully recovered.

### B. Security analysis

First, we will make some brief expression to streamline the proof. According to [32], $[p]_1$ represents $g_1^p$, $[y]_2$ represents $J_y^2$ and $[Q]_T$ represents $e(g, h)$. For a column vector $u := (u_1, \ldots, u_n)^T$, $[u]_1$ is a n-dimensional tuple $(g_1^{u_1}, \ldots, g_n^{u_n})^T$. It is similar for a matrix $N$, And for two matrices $E, F, [E^T F]_T$ denotes $e([E]_1, [F]_2)$. The outputs of Samp($\varphi$) is

$$Q := \begin{bmatrix}\mu_1 & 0 \\ 0 & \mu_2 \\ 1 & 1\end{bmatrix}, \quad Q^\perp := \begin{bmatrix}\mu_1^{-1} \\ \mu_2^{-1} \\ -1\end{bmatrix}$$

Where $\mu_1, \mu_2 \leftarrow Q_p^*$. If we make

$$E := \begin{bmatrix}e_1 & 0 \\ 0 & e_2 \\ 1 & 1\end{bmatrix}, \quad w := \begin{bmatrix}w_1 \\ w_2\end{bmatrix}, \quad w' := \begin{bmatrix}w_1 \\ w_2 \\ w\end{bmatrix}$$

The DLIN assumption can be rewritten as

$$([E]_1, [E]_2, [Ew]_1, [Ew]_2) \approx ([E]_1, [E]_2, [w']_1, [w']_2),$$

where the symbol $\approx$ implies the former and the latter are indistinguishable.

**Theorem 1.** *If the underlying ABE [4] is a completely secure scheme under the DLIN assumption, then the proposed structure is secure.*

*Proof.* Firstly, we will give the oracles of the underlying scheme $\Pi_{und}$.

*Setup*: The challenger $C$ of $\Pi_{und}$ gets the public parameters by running the group generator, and then applies the $Samp(q)$ algorithm to gain $(E, b^\perp), (F, c^\perp)$. choose $a_1, a_2, a_3 \xleftarrow{R} Z_p^*$ and set $a = (a_1, a_2, a_3)^\perp$ be a column vector. Eventually, it returns $pk := ([E]_2, [a^T E]_T)$ and $msk := (pp, E, F[a]_1)$.

*Key query*: When the adversary $A$ of $\Pi_{und}$ makes a query to this oracle. The challenger $C$ simulates the random oracle by using two lists $Lt_1$ and $Lt_2$. $Lt_1$'s entries are made up of $(x, W_x)$ or $(j, U_j)$ where $x \in \{0,1\}^*$ and $j \xleftarrow{R} Q_p^*$, and $W_x, U_j$ are $3 \times 3$ matrices over $Q_p$. And $(c, p)$ combine to generate $Lt_2$'s entries. where $c$ represents the query that $A$ will make, and $p$ belongs to $G$. When $A$ has a query of $xnt$, for $n \in \{1, 2, 3\}$ and $t \in \{1, 2\}$, $C$ first examines whether $Lt_2$ has already been queried for the query $(xnt, p)$. If the inquiry is present, $C$ outputs $p$, or $C$ examines whether $Lt_1$ contains $(x, W_x)$. If yes, $C$ wil compute $p := [(W_x^T E)_{n,t}]_1$, then outputs $p$ and adds $(xnt, p)$ to $Lt_2$. Or it chooses a random $3\times3$ matrices $W_x$ and adds $(x, W_x)$ to $Lt_1$, then $C$ computes $p$ as the former example and adds $(xnt, p)$ to $Lt_2$. Eventually, $p$ is given to $A$. When the query is $0xnt$, $C$ examines whether $Lt_2$ exists the query $(0jnt, p)$. If the query is present, $C$ outputs $p$, or $C$ examines whether $(j, U_j)$ in $Lt_1$. If yes, $C$ will compute $r := [(U_j^T E)_{n,t}]_1$, then adds $(0jnt, p)$ to $Lt_2$ and outputs $p$. Or it chooses a random $3 \times 3$ matrices $U_j$ and adds $(j, U_j)$ to $Lt_1$, then $C$ adds $(0jnt, p)$ to $Lt_2$ after computing $p$ as in the preceding case. Eventually, $p$ is provided to $A$. In all other cases, $C$ checks to see if $(c, p)$ has been questioned in $Lt_2$. If so, $C$ produces the value $p$. Alternatively, $C$ selects $p_0 \in G$ and adds $(c, p_0)$ to $Lt_2$. Eventually, $A$ receives $p_0$. After gaining a key query $Rq$ from $A$, $C$ first examines whether the query has existed. For each $y \in Q$, if $Lt_1$ does not exist $(y, W_y)$ or $U_1$, then $C$ generates them in the same way. Or $C$ computes $sk_0 = [Br]_2, sk_y = [W_y Br + \sigma_y a^\perp]$, and $sk' = [d + U_1 Br + \sigma' a^\perp]_1$, where $r_1, r_2, \sigma', \sigma_y$ are randomly chosen from $Z_p$, and $r$ represents a 2-dimensional vector $(r_1, r_2)^T$. Eventually, $A$ receives $(sk_0, \{sk_y\}_{y\in S}, sk')$.

*Encryption query*: When $C$ gains a message $msg$ and an access policy $(N, \rho)$ from $A$, $C$ initially examines whether the query has existed. If $Lt_2$ does not exist $[(W_{\rho(t)}^T E)_{n,z}]_1$ or $[(U_j^T E)_{n,z}]_1$, $C$ constructs them in the same manner. Or $C$ outputs $ct_0 = [E]_2, ct_t = [(w_{\rho(t)}^T E + \sum_{j=1}^{m_2}(E)_{t,j} U_j^T E]_1$ and $ct' = [a^T E]_T \cdot msg$, where $p_1$ and $p_2$ are randomly chosen from $Q_p$, and $p$ represents a 2-dimensional vector $(p_1, p_2)^T$. Eventually, $A$ receives $CT := (ct_0, \{ct_t\}_{t=1,\ldots,m_1}, ct')$.

*Challenge*: $A$ submits a pair of messages $(msg_1, msg_2)$, the challenger $C$ chooses a random bit $b_i$ and executes the encryption process to gain a ciphertext $CT'$. Eventually, $A$

returns a bit $b'_i$. The adversary wins the game if $b'_i = b_i$. And the advantage of $A$ to win the game is

$$Adv^{A}_{\Pi_{und}}(\varphi) \leq (8R+2)Adv^{F}_{DLIN}(\varphi) + (16R+6)/q$$

where $q = \vartheta(\varphi)$ is the order of the pairing group.

Suppose there is an adversary $E$ who can break the proposed scheme $\Pi_p$. We can construct a simulator $A_0$ to break the underlying scheme $\Pi_{und}$ with the help of $E$. The interactions between the adversary $E$ and the simulator $A_0$ are described as follows:

**Init**: The adversary $E$ submits an access structure $(\mathcal{M}^*, \rho^*)$ to $A_0$ to start the interaction.

**Setup**: The simulator obtains an instance of the underlying scheme $\Pi_{und}$ and gets the public parameter $pk$ from the challenger $C$ and returns it to $E$.

**Keygen**: The simulator requests the key from $C$ after receiving the key generation query from $E$. If the queried attribute set satisfies the challenge structure, the simulator rejects the query. Otherwise, it adds the query into his key list and then forwards the secret key $(sk_0, \{sk_y\}_{y\in S}, sk')$ to $E$.

**Encrypt**: The simulator requests the ciphertext from $C$ after receiving the encryption query from $E$. The simulator receives $CT := (ct_0, \{ct_t\}_{t=1,\ldots,m_1}, ct')$ and returns it to the adversary $E$. When $A_0$ gains a $CT$ and a key $KEK_i$, $A_0$ examines whether the *Encryption* query has existed. If it does not exists, $A_0$ firstly requests the *Encryption* query and gets $CT := (ct_0, \{ct_t\}_{t=1,\ldots,m_1}, ct')$. Then $A_0$ outputs $ct'_0 = ct_0, ct'_t = (ct_t^{KEK_i}$ and $ct'' = ct'$. Eventually, $CT' := (ct'_0, \{ct'_t\}_{t=1,\ldots,m_1}, ct'')$ is given to $E$.

**CT-update**: When $A_0$ receives a query on $CT$, he then examines whether the encryption query has been existed. If $CT$ does not exist, then $A$ obtains them from the encryption oracle. Or $C$ computes $ct'_0 = ct_0[E']_2, ct'_t = (ct_t[w^T_{\rho(t)}E' + \sum_{j=1}^{m_2}(E')_{t,j}U_j^T E']_1)^{KEK'_i}$, and $ct'' = ct'[a^T E']_T$, Eventually, $A_0$ returns $(ct'_0, \{ct'_t\}_{t=1,\ldots,m_1}, ct'')$ to $E$.

**Challenge**: $E$ submits a pair of messages $(msg_0, msg_1)$ to $A_0$, and $A_0$ forward them to the challenger of $\Pi_{und}$. Then $A_0$ forwards the ciphertext $CT_b$ to $E$, $E$ submits a bit $b'$ to $A_0$. $A_0$ returns 1 if $E$ succeeds, or 0. We suppose that all query oracles are random oracles, and $E's$ view runs with the the proposed scheme is the same as a subroutine of $A_0$. The $E$ views in $A_0$ have the same distribution. Therefore,

$$|Pr|\Pi^E_p = 1| - Pr|\Pi^{A_0}_{und} = 1|| < neg(\lambda)$$

Since the underlying scheme $\Pi_{und}$ is secure, and the advantage of an adversary $A$ to break $\Pi_{und}$ is negligible. The advantage of an adversary to break the proposed scheme $\Pi_p$ is $Adv^E_{\Pi_p}(\lambda) \approx Adv^E_{\Pi_p}(\lambda) + neg(\lambda)$. We can conclude that no PPT adversary can break the proposed scheme with a non-negligible advantage and the proposed scheme is secure.

## VII. Performance Evaluation

In this section, we will conduct a theoretical analysis to evaluate the computational and communication overhead of our scheme and in comparison to other schemes.

TABLE II
Comparison of security between schemes

| Scheme | Access policy | Revocation | EDoS attack |
|---|---|---|---|
| Xue et al. [7] | LSSS | No | Yes |
| Bayat et al. [14] | access tree | Yes | No |
| Yu et al. [19] | LSSS | Yes | No |
| Ge et al. [26] | access tree | Yes | No |
| Our scheme | LSSS | Yes | Yes |

### A. Function comparison

Table II compares access policies, revocation and protection against EDoS attacks with other schemes. Xue et al.'s scheme [7] is resistant to EDoS attacks and works for most ABE scheme, but not for some systems with revocation requirements. Yu et al.'s [19] scheme and Ge et al.'s [26] scheme implement the function of revocation, but their scheme cannot defend against EDoS attacks. The Bayat et al.'s [14] scheme implements the function of revocation, and their scheme is resistant to denial of sustainability attacks (DoS) rather than EDoS attacks. It is not listed in the table for ease of comparison with ours. Compared to these schemes, only our scheme can satisfy the requirements of revocation and protection against EDoS attacks.

### B. Theoretical Analysis

Since the system is inspired by [7], we initially conduct a theoretical comparative analysis of this system and the bottom CP-ABE system proposed by Xue. To be fair, Xue's scheme employs the same underlying ABE approach when comparing performance. [14] also proposes mechanisms to defend against DoS attacks, which we consider in our comparison. Similarly, [26] implements a revocation function, which we also take into account. Here, we do not distinguish between the operating times of elements within symmetric and asymmetric groups. $e_1$, $e_2$ and $e_t$ represent the power operations of the $G$, $H$ and $G_T$ groups, respectively, while $p$ denotes the pairing operation. The variable $T$ denotes the number of properties generated by the key, and $n_1$ refers to the number of rows in the access matrix $N$. $|G|$, $|H|$, and $|G_T|$ represent the sizes of the group $G$, $H$, and $G_T$, respectively. When updating the ciphertext, we assume that one attribute of a malicious user is revoked in the system.

Table III displays a comparison of the computational costs. Compared with the [14], our scheme significantly reduces the overhead in the decryption step. This is because the generation and encryption of the download request in our scheme employs symmetric encryption. Although there is no direct comparison listed here, subsequent experiments clearly demonstrate that our scheme exhibits substantial time overhead advantages in the generation of the download request. Theoretical analysis further indicates that our scheme slightly reduces computational overhead while enhancing under the functionalities of Xue's scheme. Compared with the scheme of [26], it can be seen that our scheme also has obvious advantages.

TABLE III
COMPARISONS OF COMPUTATION OVERHEAD

| Schemes | Keygen | Encrypt | Decrypt | Update |
|---|---|---|---|---|
| Xue et al. [7] | $(9T+12)e_1+3e_2$ | $6n_1T(n_2+1)e_1+3e_2+2e_t$ | $6p$ | |
| Bayat et al. [14] | $(3T+1)e_1$ | $2(T+1)p+(2T+1)e_1+e_t$ | $(2+T)e_1+9p$ | |
| Ge et al. [26] | $(3+T)e_t$ | $2[p+(3T+2)e_1+e_t]$ | $2[(2T+1)p+(5T+3)e_1+2Te_t]$ | $2(2e_1+3Te_1)$ |
| Our | $(9T+12)e_1+3e_2$ | $3T(2n_1n_2+2n_1+1)e_1+3e_2+2e_t$ | 6p | $3e_1(2n_1n_2+2n_1+1)+3e_2+2e_t$ |

TABLE IV
COMPARISONS OF COMMUNICATION OVERHEAD

| Schemes | User key size | Ciphertext size |
|---|---|---|
| Xue et al. [7] | 3(T+2)$\|G\|$ | $3n_1\|G\|$+3$\|H\|$+$\|G_T\|$ |
| Bayat et al. [14] | (2T+1)$\|G\|$+T$\|H\|$ | 2T$\|G\|$+$\|H\|$+$\|G_T\|$ |
| Ge et al. [26] | 5T$\|G\|$ | 2[(2T+1)$\|G\|+\|G_T\|$] |
| Our | 3(T+2)$\|G\|$ | $3n_1\|G\|$+3$\|H\|$+$\|G_T\|$ |

TABLE V
GAS COST

| Step | Verify |
|---|---|
| Deploy | 55386 |
| Total | 230275 |

Table IV presents a comparison of the communication overhead for our scheme with other related works. Before analyzing the data, it is important to note that in the MNT224 curve, the size of the element on group $H$ is three times larger than that on group $G$. our proposed scheme has the smallest number of elements on group $H$. Compared with the Xue's scheme, because we use the same encryption algorithm, the size of ciphertext and key is the same. As we can see from Table IV, as the properties increase, the keys and ciphertext sizes of [14] and [26] increase more. In this case, our scheme has a more obvious advantage.

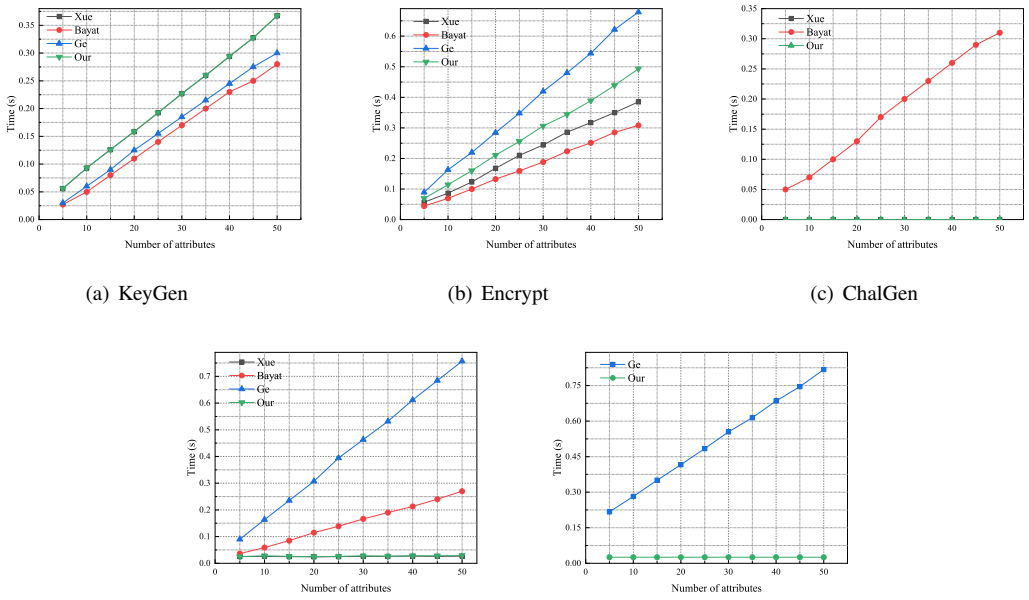
### C. Experimental Analysis

To validate our proposed scheme, we use the charm 0.50 framework in Python 3.6 to conduct a series of experimental analyses on a laptop with Intel(R) Core(TM) i5-10500 CPU @ 3.10GHz and 12 GB of memory, running Ubuntu 20.04. We compare our results with other relevant schemes. We employ Remix to generate a test version of our smart contract. The specific configuration used is the compiler (0.4.19+commit.c4cbbb05), written in solidity, with deployment on the EVM version (compiler default). In particular, due to serious security concerns with symmetrical bilinear pairing [33], we use the MNT224 curve for pairing, which offers a security level of 112-bits. The elements in group $G, H$ and $G_T$ each require 224 bits, 672 bits and 1344 bits, separately. The complexity of access policies affects the computational cost of these scenarios. We establish the access policy encompassing $B'_1$ and $B'_2$ and... and $B'_m$, ensuring all $m$ properties are included within this access policy. Given the accuracy of the experimental analysis, in the experiment, we assume that an attribute of a data user is revoked. We calculate the run time in the experiment by averaging the results of ten executions of each operation. The experimental results are shown in the Fig. 3.

- **KeyGen Time:** Fig. 3a depicts how our scheme compares to the time of key generation with other schemes. The schemes of Bayat and Ge use the Waters' underlying encryption algorithm. Our scheme and Xue's scheme use the underlying encryption algorithm of FAME. The time overhead of these schemes in the key generation phase increases with the increase of attributes. It shows that other schemes perform well in the key generation phase. Our scenario does not have an superiority in the time overhead of the key generation phase. But under the standard assumption it is completely secure, which leads to better security. Additionally, it is also acceptable for central authority, and the time of the ABE algorithm raises linearly as the amount of property raises.
- **Encrypt Time:** Fig. 3b shows how this scheme compares with other schemes in the encryption phase. It can be seen that the time cost of both our scheme and the comparison scheme in the encryption phase increases with the increase of the number of attributes. Bayat's scheme has the lowest time overhead, and Ge's scheme has the highest time overhead, because their scheme include an integrity verification process. The time cost of our scheme at this stage is higher than that of Xue's scheme, because our scheme also implements revocation.
- **ChalGen Time:** Fig. 3c depicts the scheme in this paper and the Xue and Bayat's scheme in the time cost of validating the user's download request. It can be seen that Bayat's scheme is expensive at this stage, because the challenge of this scheme is to encrypt the data. The challenge of our scheme and Xue's scheme is random numbers, so it is less expensive.
- **Decrypt Time:** The overhead of this phase in the decryption step is depicted in Fig. 3d compared to other schemes. It can be seen that this scenario has a very obvious advantage in the decryption phase. This is because our scheme uses the underlying encryption algorithm of FAME, and the most prominent advantage of this algorithm is that it is fast decryption. The decryption cost of our scheme only needs 0.02s, so the blockchain can determine very quickly whether the data user can download the file. From the figure, we can also see

This article has been accepted for publication in IEEE Transactions on Computers. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TC.2024.3398502

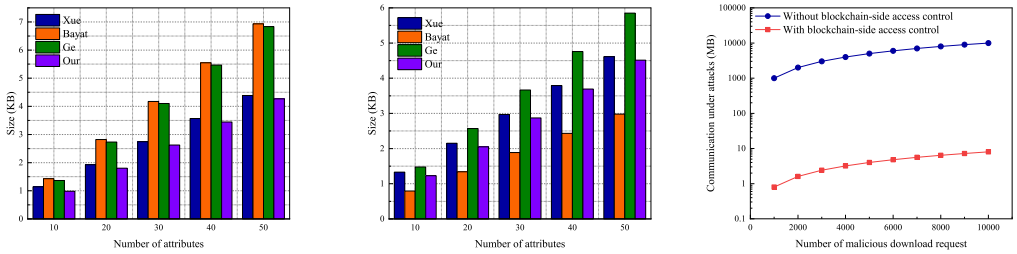





(a) KeyGen (b) Encrypt (c) ChalGen

(d) Decrypt (e) Update

Fig. 3. Experimental results in terms of computational overhead.




(a) User Key Size (b) Ciphertext Size (c) Communication under EDoS attacks

Fig. 4. Experimental results in terms of communication overhead.

that the time overhead of Ge's scenario decryption is comparatively high because their scheme has an integrity verification process in the decryption step.

- **Update Time:** Fig. 3e shows the overhead of updating ciphertexts for our scheme and Ge's scheme. As we can see, when we revoke one of the data user's attributes, we have a very clear advantage over the update overhead. This is because, in our scenario, we need to update the ciphertext related to the revocation of attributes. However, Ge's scheme requires updating all ciphertext.
- **Gas Cost:** Our verification of the data user's download request is performed with the help of the blockchain. Table 5 lists the gas costs of running this scheme. It mainly lists the deployment contract and the total gas cost.

In addition, we analyze the communication overhead of these scenarios.

- **User Key Size:** Fig. 4a depicts contrast of our scenario with other scenarios in user key size. The results show that Xue and our scenario have significant advantages over the other two scenarios in key size. In addition, although our scheme uses the same underlying encryption algorithm as Xue's scheme, Xue's scheme includes some additional verification algorithms. Therefore, the cost is higher than our scheme.
- **Ciphertext Size:** Fig. 4b shows that a comparison of our scheme with other schemes in ciphertext size. Ge's scheme is the most expensive, because it includes an integrity verification process. Although our scheme uses the same underlying encryption algorithm as Xue's scheme, Xue's scheme includes some additional verification algorithms. Therefore, the cost is higher than our scheme. As you can see in the figure, Bayat's scheme has lower overhead, but our scheme has better security and has more features.
- **Communication Under EDoS Attacks:** We set the amount of attributes per download request to 50 and the file size to 1MB. Fig. 4c shows that our proposed system significantly reduces the communication overhead under EDoS attacks.

To sum up, compared with [7], our proposed system realizes more functions under the access control function of download requests. It can be adapted to the CP-ABE system with more requirements. It doesn't incur significant overhead.

## VIII. Conclusion

EDoS attacks cannot be ignored in cloud-based data sharing. To address this issue, this study introduces blockchain technology and proposes a fine-grained access control scheme based on such technology. With our system, the blockchain is a trusted institution that determines whether the data user has permission to download. This alleviates EDoS attacks launched by malicious data users on the cloud and reduces the additional overhead caused by collusion with the malicious users on the data owner side of the semi-trusted cloud provider. Based on symmetric encryption technology and updated ciphertext technology, our scheme applies a new mechanism to solve the problem of a malicious data user revocation under an EDoS attack. A formal security certificate verifies the security of the proposed scheme. The results of our experiment show that the proposed system does not incur any significant computational or communication overhead compared with other existing systems.

## Acknowledgments

The work was supported in part by the National Natural Science Foundation of China under Grant 62272002, Grant 62202005, Grant 62372002 and Grant U23A20308, in part by the Natural Science Foundation of Anhui Province, China under Grant 2208085QF198 and in part by the University Synergy Innovation Program of Anhui Province under Grant GXXT-2022-049. The authors are very grateful to the anonymous referees for their detailed comments and suggestions regarding this paper.

## References

[1] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: state-of-the-art and research challenges," *Journal of internet services and applications*, vol. 1, no. 1, pp. 7–18, 2010.

[2] J. Cui, B. Li, H. Zhong, G. Min, Y. Xu, and L. Liu, "A practical and efficient bidirectional access control scheme for cloud-edge data sharing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 2, pp. 476–488, 2022.

[3] Z. Zhang, W. Zhang, and Z. Qin, "Fully constant-size cp-abe with privacy-preserving outsourced decryption for lightweight devices in cloud-assisted iot," *Security and Communication Networks*, vol. 2021, 2021.

[4] S. Agrawal and M. Chase, "Fame: fast attribute-based message encryption," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 665–682.

[5] S. Potluri, M. Mangla, S. Satpathy, and S. N. Mohanty, "Detection and prevention mechanisms for ddos attack in cloud computing environment," in *2020 11th international conference on computing, communication and networking technologies (ICCCNT)*. IEEE, 2020, pp. 1–6.

[6] M. A. S. Monge, J. M. Vidal, and G. M. Pérez, "Detection of economic denial of sustainability (edos) threats in self-organizing networks," *Computer Communications*, vol. 145, pp. 284–308, 2019.

[7] K. Xue, W. Chen, W. Li, J. Hong, and P. Hong, "Combining data owner-side and cloud-side access control for encrypted cloud storage," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 8, pp. 2062–2074, 2018.

[8] J. Ning, X. Huang, W. Susilo, K. Liang, X. Liu, and Y. Zhang, "Dual access control for cloud-based data storage and sharing," *IEEE Transactions on Dependable and Secure Computing*, vol. 19, no. 2, pp. 1036–1048, 2020.

[9] K. Y. Yap, H. H. Chin, and J. J. Klemeš, "Blockchain technology for distributed generation: A review of current development, challenges and future prospect," *Renewable and Sustainable Energy Reviews*, vol. 175, p. 113170, 2023.

[10] C. Yuan, M. Xu, X. Si, and B. Li, "Blockchain with accountable cp-abe: how to effectively protect the electronic documents," in *2017 IEEE 23rd international conference on parallel and distributed systems (ICPADS)*. IEEE, 2017, pp. 800–803.

[11] Y. Rahulamathavan, R. C.-W. Phan, M. Rajarajan, S. Misra, and A. Kondoz, "Privacy-preserving blockchain based iot ecosystem using attribute-based encryption," in *2017 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*. IEEE, 2017, pp. 1–6.

[12] H. Cui, R. H. Deng, Y. Li, and B. Qin, "Server-aided revocable attribute-based encryption," in *Computer Security–ESORICS 2016: 21st European Symposium on Research in Computer Security, Heraklion, Greece, September 26-30, 2016, Proceedings, Part II 21*. Springer, 2016, pp. 570–587.

[13] J. Ning, X. Dong, Z. Cao, L. Wei, and X. Lin, "White-box traceable ciphertext-policy attribute-based encryption supporting flexible attributes," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 6, pp. 1274–1288, 2015.

[14] M. Bayat, H. R. Arkian, and M. R. Aref, "A revocable attribute based data sharing scheme resilient to dos attacks in smart grid," *Wireless Networks*, vol. 21, no. 3, pp. 871–881, 2015.

[15] J. Idziorek, M. Tannian, and D. Jacobson, "Attribution of fraudulent resource consumption in the cloud," in *2012 IEEE fifth international conference on cloud computing*. IEEE, 2012, pp. 99–106.

[16] M. H. Sqalli, F. Al-Haidari, and K. Salah, "Edos-shield-a two-steps mitigation technique against edos attacks in cloud computing," in *2011 Fourth IEEE international conference on utility and cloud computing*. IEEE, 2011, pp. 49–56.

[17] S. Ding, J. Cao, C. Li, K. Fan, and H. Li, "A novel attribute-based access control scheme using blockchain for iot," *IEEE Access*, vol. 7, pp. 38 431–38 441, 2019.

[18] A. Z. Ourad, B. Belgacem, and K. Salah, "Using blockchain for iot access control and authentication management," in *Internet of Things–ICIOT 2018: Third International Conference, Held as Part of the Services Conference Federation, SCF 2018, Seattle, WA, USA, June 25-30, 2018, Proceedings 3*. Springer, 2018, pp. 150–164.

[19] K. Yu, L. Tan, M. Aloqaily, H. Yang, and Y. Jararweh, "Blockchain-enhanced data sharing with traceable and direct revocation in iiot," *IEEE transactions on industrial informatics*, vol. 17, no. 11, pp. 7669–7678, 2021.

[20] J. Wan, J. Li, M. Imran, D. Li *et al.*, "A blockchain-based solution for enhancing security and privacy in smart factory," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3652–3660, 2019.

[21] J. Huang, L. Kong, G. Chen, M.-Y. Wu, X. Liu, and P. Zeng, "Towards secure industrial iot: Blockchain system with credit-based consensus mechanism," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 6, pp. 3680–3689, 2019.

[22] C. Lee, S. Zhang, and K. Ng, "Development of an industrial internet of things suite for smart factory towards re-industrialization," *Advances in manufacturing*, vol. 5, no. 4, pp. 335–343, 2017.

[23] J. Li, Y. Shi, and Y. Zhang, "Searchable ciphertext-policy attribute-based encryption with revocation in cloud storage," *International Journal of Communication Systems*, vol. 30, no. 1, p. e2942, 2017.

[24] X. Yan and H. Meng, "Ciphertext policy attribute-based encryption scheme supporting direct revocation," *Journal on Communications*, vol. 37, no. 5, pp. 44–50, 2016.

[25] C. Bai, Y. Zhang, H. Ma, and Z. Liu, "Expressive ciphertext-policy attribute-based encryption with direct user revocation," *International Journal of Embedded Systems*, vol. 9, no. 6, pp. 495–504, 2017.

[26] C. Ge, W. Susilo, J. Baek, Z. Liu, J. Xia, and L. Fang, "Revocable attribute-based encryption with data integrity in clouds," *IEEE Transactions on Dependable and Secure Computing*, 2021.

[27] J. Hur and D. K. Noh, "Attribute-based access control with efficient revocation in data outsourcing systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 7, pp. 1214–1221, 2010.

[28] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security*, 2006, pp. 89–98.

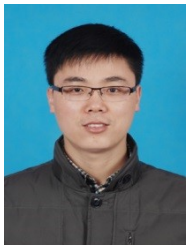
[29] L. Luu, D.-H. Chu, H. Olickel, P. Saxena, and A. Hobor, "Making smart contracts smarter," in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 254–269.
[30] D. Han, N. Pan, and K.-C. Li, "A traceable and revocable ciphertext-policy attribute-based encryption scheme based on privacy protection," *IEEE Transactions on Dependable and Secure Computing*, 2020.
[31] D. A. McGrew and J. Viega, "The security and performance of the galois/counter mode (gcm) of operation," in *International Conference on Cryptology in India*. Springer, 2004, pp. 343–355.
[32] J. Chen, R. Gay, and H. Wee, "Improved dual system abe in prime-order groups via predicate encodings," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2015, pp. 595–624.
[33] C.-I. Fan, V. S.-M. Huang, and H.-M. Ruan, "Arbitrary-state attribute-based encryption with dynamic membership," *IEEE Transactions on Computers*, vol. 63, no. 8, pp. 1951–1961, 2013.

**Qingyang Zhang** was born in Anhui Province, China, in 1992. He received his B. Eng. degree and Ph.D. degree in computer science from Anhui University in 2014 and 2021, respectively. He is currently an associate professor of School of Computer Science and Technology at Anhui University. His research interest includes edge computing, computer systems, and security. He has over 30 scientific publications in reputable journals (e.g. Proceedings of the IEEE, IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Computers) and international conferences.

**Chang Xu** is now a research student in the School of Computer Science and Technology, Anhui University. Her research focuses on the security of cloud storage.

**Hong Zhong** was born in Anhui Province, China, in 1965. She received her PhD degree in computer science from University of Science and Technology of China in 2005. She is currently a professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. Her research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). She has over 200 scientific publications in reputable journals (e.g. IEEE Journal on Selected Areas in Communications, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Mobile Computing, IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Multimedia, IEEE Transactions on Vehicular Technology, IEEE Transactions on Network and Service Management, IEEE Transactions on Cloud Computing, IEEE Transactions on Industrial Informatics, IEEE Transactions on Industrial Electronics and IEEE Transactions on Big Data), academic books and international conferences.

**Chengjie Gu** received his Ph.D. degree from the Nanjing University of Posts and Telecommunications in 2012. From 2012 to 2017, he was an innovation team leader at the 38th Research Institute of CETC and conducted research and development in the communication and networking sector. Currently, he is a dean of the School of public security and Emergency of Anhui University of Science and Technology. He has completed postdoctoral research at the USTC. He is a high-level innovation leader in Anhui province and a cybersecurity expert in Zhejiang province in China. His research interest includes network security and trusted network architecture, etc.



**Jie Cui** (Senior Member, IEEE) was born in Henan Province, China, in 1980. He received his Ph.D. degree in University of Science and Technology of China in 2012. He is currently a professor and Ph.D. supervisor of the School of Computer Science and Technology at Anhui University. His current research interests include applied cryptography, IoT security, vehicular ad hoc network, cloud computing security and software-defined networking (SDN). He has over 150 scientific publications in reputable journals (e.g. IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Information Forensics and Security, IEEE Journal on Selected Areas in Communications, IEEE Transactions on Mobile Computing, IEEE Transactions on Parallel and Distributed Systems, IEEE Transactions on Computers, IEEE Transactions on Intelligent Transportation Systems, IEEE Transactions on Network and Service Management, IEEE Transactions on Industrial Informatics, IEEE Transactions on Industrial Electronics, IEEE Transactions on Cloud Computing and IEEE Transactions on Multimedia), academic books and international conferences. He is in the Editorial Board of several international journals, such as IET Communications, Security and Communication Networks, and Sensors.