Data Sharing in the Metaverse With Key Abuse Resistance Based on Decentralized CP-ABE

Liang Zhang ⬚, Zhanrong Ou ⬚, Changhui Hu ⬚, Haibin Kan ⬚, *Member, IEEE*, and Jiheng Zhang ⬚

Abstract—Data sharing is ubiquitous in the metaverse, which adopts blockchain as its foundation. Blockchain is employed because it enables data transparency, achieves tamper resistance, and supports smart contracts. However, securely sharing data based on blockchain necessitates further consideration. Ciphertext-policy attribute-based encryption (CP-ABE) is a promising primitive to provide confidentiality and fine-grained access control. Nonetheless, authority accountability and key abuse are critical issues that practical applications must address. Few studies have considered CP-ABE key confidentiality and authority accountability simultaneously. To our knowledge, we are the first to fill this gap by integrating non-interactive zero-knowledge (NIZK) proofs into CP-ABE keys and outsourcing the verification process to a smart contract. To meet the decentralization requirement, we incorporate a decentralized CP-ABE scheme into the proposed data sharing system. Additionally, we provide an implementation based on smart contract to determine whether an access control policy is satisfied by a set of CP-ABE keys. We also introduce an open incentive mechanism to encourage honest participation in data sharing. Hence, the key abuse issue is resolved through the NIZK proof and the incentive mechanism. We provide a theoretical analysis and conduct comprehensive experiments to demonstrate the feasibility and efficiency of the data sharing system. Based on the proposed accountable approach, we further illustrate an application in GameFi, where players can play to earn or contribute to an accountable DAO, fostering a thriving metaverse ecosystem.

Index Terms—Metaverse, accountability, CP-ABE, key abuse, GameFi, DAO.

I. INTRODUCTION

THE metaverse is going to mix virtual and real worlds more tightly [1]. In the metaverse, users need to frequently exchange information to enable collaboration across enterprises or among individuals. Traditional data sharing approaches [2], [3], [4] rely on a central server or an authority to accomplish data exchange. However, this leads to a single point of failure. It is widely recognized that blockchain is a suitable foundation to build connection between virtual and real worlds. GameFi [43] exemplifies a core component of the metaverse, showcasing how blockchain can transform digital experiences. GameFi platforms give players greater control over their assets and data, such as in-game items like weapons and skins. Additionally, GameFi operates within its own economy, allowing users to buy, sell, and trade virtual goods and services. This ecosystem enhances engagement through rewards, challenges, and achievements, making the gaming experience more immersive and rewarding.

Blockchain is seen as a crucial technology to provide a trustworthy approach to store and exchange data. Also, blockchain is the most friendly platform to host smart contract. Previous data sharing systems [6], [7] have shown that blockchain can help enhance convenience and credibility. In data sharing, blockchain can be adopted as a data registry or data repository [5], [8]. Besides, blockchain's smart contract execution engine can be regarded as a trustworthy verifier [9]. However, there are obstacles when designing data access control on a blockchain due to its transparency, which may bring disastrous risks to data security. The primary security risks include privacy leakage and unauthorized access.

Ciphertext-policy attribute-based encryption (CP-ABE) [10], [11], [12] is a promising cryptographic primitive to provide both confidentiality and access control simultaneously. In CP-ABE, the ciphertext is embedded with an access control policy and the decryption key is generated with an attribute set. Only when the set satisfies the policy can the ciphertext be decrypted. Thus, encryptors can protect data privacy using proper access control policy, without caring about the identities of decryptors. That is to say, CP-ABE allows a data owner to share data in a one-to-many paradigm. By incorporating an access control policy in a ciphertext, CP-ABE enables fine-grained access control. In addition, multi-authority CP-ABE [11], [12], also

known as decentralized CP-ABE, allows multiple distributed attribute authorities to generate keys autonomously and jointly in decentralized applications.

However, CP-ABE keys are sensitive information that should be distributed securely and efficiently to users. Key abuse mainly includes illegal authorization and key disclosure [13]. Illegal authorization means that dishonest authorities may generate keys for some unauthorized data users. Key disclosure implies that a dishonest data user may intentionally expose the decryption keys for profit or other illegal usage, leading to risks such as impersonation attacks or unauthorized decryption. Researches [15], [16], [17] investigate methods to track the user who leaks the key by mapping the key to the user. However, these schemes require suspected keys to be exposed for verification or audition.

To further uncover misbehavior of authorities, some CP-ABE schemes [15], [16], [17], [20], [22], [41] are dedicated to achieving the property of auditability or accountability. Accountability means that malicious authorities should be caught if they issue illegal keys to authorized users or legitimate keys to unauthorized users. A common approach [15], [16], [17], [20], [41] is to define an algorithm, which inputs a decryption key and judges whether it is well-formed or finds out its owner. However, there are major limitations in practice. On one hand, decryption keys should be exposed if they need to be audited. On the other hand, dishonest users may hide decryption keys as if they have not received the keys. Some other works employ blockchain to gain public auditability and use encryption technology to preserve the privacy of CP-ABE keys [23], [42]. Also, an incentive mechanism based on blockchain is adopted to implement key traceability [13].

In this paper, we envision a data sharing framework for the metaverse with accountable authorities applying a decentralized CP-ABE scheme [11]. We depict the data sharing paradigm as below. Firstly, a data user, who wants to access an owner's data, deposits some digital currency in a smart contract. Then, attribute authorities generate CP-ABE keys and encrypt them with the user's public key. Also, NIZK proofs are attached. Next, authorities publish the encrypted CP-ABE keys and NIZK proofs on a blockchain, where the encrypted keys are recorded and verified. Meanwhile, the owner and honest authorities will be rewarded by partitioning the deposits. Thereafter, the user obtains a set of valid CP-ABE keys using its private key. Finally, the user can acquire the owner's shared data. We further apply the proposed framework in GameFi, enabling players (or providers) to sell (or issue) in-game items with flexible access control. This approach provides a transparent and secure system for players to engage in play-to-earn activities with accountable DAOs.

Our contributions of this paper are summarized as below:

1) We propose a secure and efficient blockchain-driven data sharing framework for the metaverse leveraging decentralized CP-ABE, guaranteeing data confidentiality with fine-grained access control.

2) We are the first to consider CP-ABE key privacy and accountability simultaneously based on blockchain. Non-interactive zero-knowledge (NIZK) proofs are generated

to guarantee accountability and they are automatically verified through a smart contract.

3) We implement the CP-ABE access control policy in smart contract. Further, an open incentive mechanism is proposed to encourage honest behaviors. Based on the mechanism, key abuse problems and collusion attacks are eliminated.

4) We give concrete security analysis of the proposed method and conduct comprehensive experiments on the Ethereum. The experimental results show the feasibility and practicality of our proposed scheme.

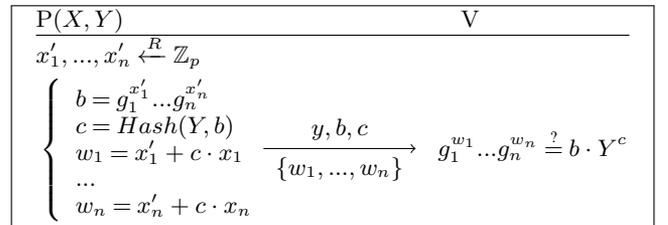## II. PRELIMINARIES

A. Bilinear Mapping

$\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ are groups of prime order p. Let g_1, g_2 be a generator of $\mathbb{G}_1$ and $\mathbb{G}_2$ repectively and $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ be a bilinear map with following properties:

1) Bilinearity: for all $\mu \in \mathbb{G}_1$, $\upsilon \in \mathbb{G}_2$ and $a, b \in \mathbb{Z}_p$, we have $e(\mu^a, \upsilon^b) = e(\mu, \upsilon)^{ab}$.

2) Nondegeneracy: $e(g_1, g_2) \neq 1$.

Usually, the operation e is efficiently computable. The resulting group $\mathbb{G}_T$ is called a bilinear mapping group.

B. Linear Relationship Sigma protocol and NIZK Proof

The Sigma protocol [29] is a three-move protocol between a prover and a verifier that can be used to prove knowledge of certain relationships without revealing any information about the knowledge itself. In linear relationship Sigma protocol, a prover P can prove zero knowledge of witness $X = \{x_1, ..., x_n\}$ for statement Y as below, where $Y = h_1^{x_1}...h_n^{x_n}$ and $h_1, ..., h_n$ are generators of $\mathbb{G}$.

$$
\begin{array}{ll}
\underline{\text{P}(X, Y)} \hspace{4cm} \text{V} \\
x_1', ..., x_n' \xleftarrow{R} \mathbb{Z}_p \\
\left\{
\begin{array}{l}
b = g_1^{x_1'}...g_n^{x_n'} \\
c = Hash(Y, b) \\
w_1 = x_1' + c \cdot x_1 \\
... \\
w_n = x_n' + c \cdot x_n
\end{array}
\right.
\xrightarrow[\{w_1, ..., w_n\}]{y, b, c}
\ g_1^{w_1}...g_n^{w_n} \overset{?}{=} b \cdot Y^c
\end{array}
$$

$Hash$ is modeled as a random oracle, as required by Fiat-Shamir heuristic [30]. Denote the conversation $(b, c, \{w_1, ..., w_n\})$ as proofs for the prover P. It can be concluded that above protocol constructs non-interactive zero knowledge (NIZK) system between P and V. The required security properties of Sigma protocol are given by Definition 1 in the Appendix A.

C. Decentralized CP-ABE

Decentralized CP-ABE scheme [11], [12] contains following five algorithms:

- $GP \leftarrow \mathsf{GlobalSetup}(\Lambda)$. It takes in the security parameter Λ and outputs global parameters GP.
- $\mathsf{sk}_\theta, \mathsf{pk}_\theta \leftarrow \mathsf{ABEAuthSetup}(GP, \theta)$. Each authority θ takes GP as input to produce a key pair $(\mathsf{sk}_\theta, \mathsf{pk}_\theta)$.

$$\text{ABEKeyGen}(GID, GP, u, \text{sk}_\theta) \rightarrow (K_0, K_1):$$
$$d_\theta \xleftarrow{R} \mathbb{Z}_p$$
$$\begin{cases} K_0 = g_1^{\alpha_\theta} H(GID)^{\beta_\theta} F(u)^{d_\theta} \\ K_1 = g_2^{d_\theta} \end{cases}$$

Fig. 1. RW CP-ABE ABEKeyGen algorithm.

- $C \leftarrow \text{ABEEncrypt}(M, acp, GP, \{\text{pk}_\theta\})$. The algorithm takes in GP, a message $M \in \mathbb{G}_T$, an access control policy acp and a set of public keys. It outputs a ciphertext C.
- $K_0, K_1 \leftarrow \text{ABEKeyGen}(GID, GP, u, \text{sk}_\theta)$. The algorithm takes in an identity GID, GP, an attribute u belonging to the authority θ, and an authority's secret key sk_θ. It produces a decryption key $K_{GID,u}$. For convenience, we use K_0, K_1 and omit GID and u where RW CP-ABE is leveraged in this paper.
- $M \leftarrow \text{ABEDecrypt}(GP, C, \{K_{GID,u}\})$. The decryption algorithm takes in GP, the ciphertext C, and a collection of decryption keys. Only if the collection of decryption keys satisfies the access control policy in the ciphertext, it outputs the message M.

We concentrate on the authority accountability feature. ABEKeyGen is the interface to issue keys for decryptors. Our idea of accountability can be applied to most CP-ABE schemes, in which keys are group elements (i.e., public key-style). Without loss of generality, we choose Rouselakis and Waters's (RW) [11] decentralized CP-ABE scheme. Fig. 1 depicts the ABEKeyGen RW CP-ABE algorithm.

Where $\text{sk}_\theta = (\alpha_\theta, \beta_\theta)$ and $\text{pk}_\theta = (g_1^{\alpha_\theta}, g_2^{\alpha_\theta}, g_2^{\beta_\theta}, e(g_1, g_2)^{\alpha_\theta})$ are authority θ's key pair; g_1 and g_2 are generators of group $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively; H maps global identity GID to an element on $\mathbb{G}_1$; F maps an arbitrary string to an element on $\mathbb{G}_1$. Note that $g_1^{\alpha_\theta}$ is exposed in our application. It does not reduce the scheme security, since K_0 is also protected by β_θ and a blinded factor d_θ.

D. Ethereum Blockchain

Ethereum [31] is one of the most innovative blockchain. Its security is based on distributed storage using a cryptographic chain of blocks and the consensus algorithm. Smart contract is a program that aims at building agreement among distributed nodes. Once the contract is executed in Ethereum virtual machine (EVM), it is written onto the blockchain, making it traceable and irreversible. Each smart contract has a unique address, which is created when the user deploys the contract. Pairing-check is a cryptographic operation that judges whether multiple group elements satisfy specific group mapping defined in EIP-197 [33]. However, pairing-check only outputs a boolean value, rather than providing a group mapping (or bilinear mapping) result. The gas system is designed to eliminate distributed denial-of-service (DDoS) attacks on the Ethereum network. Every computation and storage costs a certain amount of gas (digital currency) from the transaction invoker. The used gas is packed as a reward for miners. In Ethereum, Ether is widely adopted digital currency units, and 1Ether=10^{18}wei.

III. SYSTEM MODEL AND THREAT MODEL

A. System Entities

1) **Data owner**: A data owner is the possessor of data (i.e., digital assets, such as communication keys, NFTs, files and other electronic information). Data owners share data, which is encrypted with RW CP-ABE ABEEncrypt algorithm, for digital currency as benefits.
2) **Data user**: A data user with (pk_u, sk_u) is the demanding side who wants to use data. Data users pay for the data when they are authorized to access it (i.e., they are able to decrypt the corresponding RW CP-ABE ciphertexts).
3) **Attribute Authorities (AAs)**: Each authority θ with $(\text{pk}_\theta, \text{sk}_\theta)$ earns digital currency by offering key issuance services for data users. Authorities are independently and autonomously distributed all over the world.
4) **Blockchain**: Blockchain is used as a decentralized, available and tamper-resistant service, providing reliable storage and verifiable computation. In our proposed system, it is used to manage (i.e., store and verify) RW CP-ABE keys. Particularly, we choose Ethereum, a permissionless ecosystem, as the underlying blockchain.
5) **Data storage provider (DSP)**: Data storage provider is the service to store CP-ABE ciphertext from data owners.

B. System Assumption and Threats

We aim at achieving a blockchain and CP-ABE based privacy-preserving data sharing framework in the metaverse. Specifically, CP-ABE keys are managed transparently yet accountably on the blockchain and key abuse resistance should be considered. Global identifier GID is unique to each sharing instance. An attribute string u can be used in different sharing instances.

If an entity abides by the proposed scheme, it is assumed to be honest. Otherwise, it is regarded to be dishonest or malicious. We assume an acp in a sharing instance cannot be satisfied by keys issued only by malicious authorities. Hence, we do not consider the collusion attack between user and authorities. Authorities are required to stake digital currency to offer key issuance services. If they are found to act dishonestly, their staked digital currency will be forfeited. The DSP is honest but curious, and its benefits within the system are not specified. The authorities are authenticated to **DSP**. The potential threats in our framework include:

Threat 1: Adversaries may try to construct/infer a key.

Threat 2: A malicious user may forge a key for an attribute u' which he/she does not have.

Threat 3: Malicious authority may generate invalid keys or fake proofs in key issuance.

Threat 4: Malicious users may try to obtain owner's ciphertext or digital assets by impersonation attack.

Threat 5: Malicious users may collude to obtain and decrypt a data owner's ciphertext or digital assets.
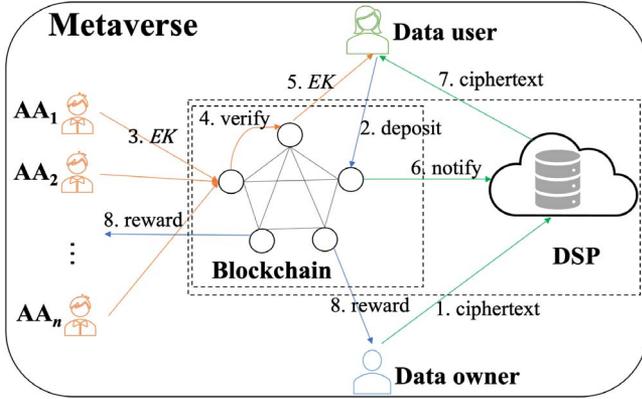
Fig. 2.   High-level overview of the system architecture.

C. System Goals

The goals of the system include the following properties.
1) *Tamper-resistance*: Data authenticity, validity, and permanence are ensured when data is stored on the chain.
2) *Decentralization*: The system is realized without a trusted third party, avoiding the single point of failure problem.
3) *Key privacy*: CP-ABE keys are encrypted and delivered through blockchain against eavesdropping attack.
4) *Data privacy*: Data is confidential to unauthorized users.
5) *Forward-secrecy*: If the CP-ABE keys of some sharing instance (GID) are compromised, the privacy of previously shared data is still guaranteed.
6) *Incentives*: Data users and authorities are rewarded for participating in the data sharing scheme.
7) *Accountability*: An encrypted key can be checked whether it is correct for the designated user with pk_u.
8) *Key abuse resistance*: Illegal key authorization is eliminated and key disclosure are useless for the system.

IV. THE PROPOSED SYSTEM

A. High-Level Overview

In the metaverse, data sharing requires no central trusted third party (TTP). Hence, blockchain is leveraged to play the role of trusted proxy and its smart contract enables parties in the metaverse to trade digital assets fairly. Moreover, we incorporate a multi-authority CP-ABE algorithm to decentralize the identities of the players or creators. Fig. 2 depicts the high-level overview of the system architecture.

Specifically, **DSP** is used to store ciphertexts, where symmetric encryption is used to wrap the CP-ABE scheme to encrypt arbitrary data. Blockchain provides tamper-proof storage and trustworthy verification computation. Attribute authorities generate CP-ABE keys and encrypt them with data user's public key. Then, they publish the encrypted CP-ABE keys to the blockchain, which records and verifies each of the encrypted keys. Finally, the data user gets valid encrypted CP-ABE keys from the blockchain and ciphertext from the **DSP** to obtain the data owner's shared data. The green arrows show the actions related to data users' ciphertext; The blue arrows demonstrate

how the incentives (digital currency) flow; The orange arrows introduce how **AA**'s encrypted keys are transferred and verified.

B. Accountability of Authorities Via Smart Contract

In CP-ABE-based applications, authorities have to use a private secure channel to deliver decryption keys after invoking the ABEKeyGen algorithm. To allow authorities to incorporate public channels, we slightly modify the ABEKeyGen algorithm to the ABEEncKey algorithm. Particularly, by replacing g_1 in K_0 with the targeted user's public key $pk_u = g_1^y$, we obtain EK_0. Hence, the encrypted keys are (EK_0, EK_1) and they can be delivered through blockchain.

$\underline{\text{ABEEncKey}(GID, GP, u, \mathsf{sk}_\theta, pk_u) \to (EK_0, EK_1):}$

$$d_\theta \xleftarrow{R} \mathbb{Z}_p$$
$$\begin{cases} EK_0 = (pk_u)^{\alpha_\theta} H(GID)^{\beta_\theta} F(u)^{d_\theta} \\ EK_1 = g_2^{d_\theta} \end{cases}$$

Obviously, the encrypted key (EK_0, EK_1) cannot be directly used to decrypt any RW CP-ABE ciphertext.

Lemma 1: The ABEEncKey algorithm is resistant to eavesdropping attacks, i.e., no adversary can recover a decryption key from eavesdropping the blockchain.

Proof: Obviously, an adversary $\mathcal{A}$ cannot learn useful information from EK_1. Neither, $\mathcal{A}$ learns nothing from EK_0 which can be regarded as a Pederson commitment [45] of the authority θ. Due to the perfect hiding of Pederson commitment, $\mathcal{A}$ cannot forge a valid commitment/ciphertext EK_0, even if the secret $(\alpha_\theta, \beta_\theta)$ are used in multiple commitments/ciphertexts, as long as $log_{pk_u} F(u)$ and $log_{H(GID)} F(u)$ are unknown. Actually, EK_0 hides K_0 using DH exchange symmetric key $(g_1^\alpha)^{y-1}$. Note that K_0 can be treated as another Pederson commitment with both g_1 and $H(GID)$ as bases, without exposing the secret $(\alpha_\theta, \beta_\theta)$ of the authority θ. Therefore, $\mathcal{A}$ cannot obtain a key given all encrypted keys are stored on publicly on blockchain, addressing Threat 1.

By proving Lemma 1, we know that only the targeted user, with secret key $sk_u = y$, can recover a valid decryption key as the getKey algorithm shows.

$\underline{\text{getKey}(EK_0, EK_1, g_1^{\alpha_\theta}, y) \to (K_0, K_1):}$

$$K_0 = EK_0/(g_1^{\alpha_\theta})^{y-1} = g_1^{\alpha_\theta} H(GID)^{\beta_\theta} F(u)^{d_\theta}$$
$$K_1 = EK_1 = g_2^{d_\theta}$$

Lemma 2: Malicious users cannot forge a key for an attribute u' which he/she does not have, addressing Threat 2.

Proof: A user who decrypts (EK_0, EK_1) and obtains key (K_0, K_1), may forge key via malleability[1] by calculating $(K_0 \cdot F(u)^r, K_1 \cdot g_2^r)$, where $r \xleftarrow{R} \mathbb{Z}_p$. Obviously, the new key is still bound to the attribute u. Hence, the user cannot successfully forge a meaningful key for attribute $u' \neq u$.

Whether an authority has correctly perform the encryption with the ABEEncKey algorithm needs to be considered, especially when authorities are beneficiaries in the metaverse applications. Therefore, we take advantage of the linear relationship

---

[1]This property sometimes is used for achieving the functionality of key delegation [10].

of the Sigma protocol (defined in Section II-B) to make the encrypted key (EK_0, EK_1) publicly verifiable. Hence, the NIZK proofs are generated as genProofs depicts.

genProofs$(GID, u, pk_u) \rightarrow$ proofs :

$$\alpha'_\theta, \beta'_\theta, d'_\theta \xleftarrow{R} \mathbb{Z}_p$$

$$\text{proofs} = \begin{cases} EK'_0 = (pk_u)^{\alpha'_\theta} H(GID)^{\beta'_\theta} F(u)^{d'_\theta} \\ EK'_1 = g_2^{d'_\theta} \\ c = Hash(EK_0, EK'_0) \\ w_1 = \alpha'_\theta + c \cdot \alpha_\theta \\ w_2 = \beta'_\theta + c \cdot \beta_\theta \\ w_3 = d'_\theta + c \cdot d_\theta \end{cases} \quad (1)$$

When $(EK_0, EK_1, \text{proofs})$ is submitted to a verifier, he/she can check whether (EK_0, EK_1) corresponds to a valid decryption key using checkKey. The checkKey algorithm inputs the sharing instance GID, the attribute u, encrypted key (EK_0, EK_1) and the NIZK proofs, and it is formally defined as below:

checkKey$(EK_0, EK_1, \text{proofs}, GID, u)$:

$$(pk_u)^{w_1} H(GID)^{w_2} F(u)^{w_3} \overset{?}{=} EK'_0 \cdot EK_0^c \quad (2)$$

$$g_2^{w_3} \overset{?}{=} EK'_1 \cdot EK_1^c \quad (3)$$

$$e(pk_u, g_2^{\alpha_\theta}) e(H(GID), g_2^{\beta_\theta}) e(F(u), EK_1) \overset{?}{=} e(EK_0, g_2) \quad (4)$$

Lemma 3: Malicious authority cannot generate invalid keys or proofs, addressing Threat 3.

Proof: The NIZK proofs in the checkKey algorithm guarantees honest issuance of (EK_0, EK_1) by the authority. The equation (2) is the linear Sigma protocol, proving knowledge of the tuple $\{\alpha_\theta, \beta_\theta, d_\theta\}$. The equation (3) is the linear Sigma protocol, proving knowledge of the $\{d_\theta\}$. Since both EK_0 and EK_1 have leveraged the same d_θ, we adopt the equation (4) to prove the binding relationship of d_θ. These equations are automatically executed on smart contract. Moreover, if any equation in checkKey does not hold, the authority is deemed dishonest, his/her stake will be forfeited as a result. The required security properties of the NIZK proof system is defined in Appendix A and the corresponding security proofs are demonstrated in the Appendix B.

C. Access Control in Smart Contract

DSP, who manages CP-ABE ciphertext, should be able to judge whether a user is genuinely granted to access the ciphertext. This is done by letting **DSP** check whether a user indeed has a legal key from authorities. A legal key is a combination of sub-keys from distributed authorities. However, checkKey defined in Section IV-B can only be used to verify each sub-key. Hence, we design a mechanism to judge the validity of a combined key in a smart contract. Suppose the combined key corresponds to the data user's attribute set $[u]$. In case a malicious user forges an attribute set, $[u]$ only contains the attribute whose corresponding sub-key is verified using checkKey. Thus, only if the data user's attribute set $[u]$ satisfies the acp, the **DSP** sends the ciphertext to the user. We omit the details on how the user (with sk_u) prove himself/herself to **DSP**.

Algorithm 1 Access control in smart contract

▷ Global variables
global $ops \leftarrow []$
global $result \leftarrow []$
function calc():
    ▷ Pop two values from $result$ and push the new result
1: **if** $result$.length < 2 **then** return
2: **end if**
3: $op = ops$.pop()
4: $t1 = result$.pop()
5: $t2 = result$.pop()
6: **if** $op ==$ "AND" **then**
7:     $result$.push($t1\&t2$)
8: **else if** $op ==$ "OR" **then**
9:     $result$.push($t1\|t2$)
10: **end if**
function judgeAttrs($[u], acp$):
    ▷ Scan acp and invoke calc to obtain the final result
1: $words \leftarrow split(acp, "\ ")$
2: **while** $word \in words$ **do**
3:     **if** $word ==$ "AND" $\|$ $word ==$ "OR" **then**
4:         **if** ops.length > 0 & ops.top() $!=$ "(" **then**
5:             calc()
6:         **end if**
7:         ops.push($word$)
8:     **else if** $word ==$ "(" **then**
9:         ops.push("(")
10:     **else if** $word ==$ ")" **then**
11:         $top \leftarrow ops$.top()
12:         **while** $top !=$ ")" **do**
13:             $top = ops$.pop()
14:             calc()
15:         **end while**
16:     **else**
17:         $result$.push($[u]$.contains(word)==true)
18:     **end if**
19: **end while**
20: **while** ops.length > 0 **do**
21:     calc()
22: **end while**
23: **Return** result[0]

The judgeAttrs function in Algorithm 1 demonstrates how to judge whether an attribute set $[u]$ satisfies the access control policy acp. Access control policy acp can be regarded as propositions, only given a proper condition (i.e., the attribute set $[u]$) can they be determined to be true. The judgeAttrs function calculates acp as an infix expression, given values $[u]$. Particularly, we use two stacks ops and $result$ to store operations (i.e., AND/OR) when scanning acp and values after calculating the operations. judgeAttrs scans the acp and invokes calc to calculate the result step by step. calc is a function to get the two top values from $result$ and it computes a new result according to the last operation from ops.

Lemma 4: Malicious users cannot obtain owner's ciphertext by impersonation attack, addressing Threat 4.

    Proof: **DSP** sends ciphertext to a user only if attributes in $[u]$ are truly used by authorities and judgeAttrs outputs true. Moreover, by Lemma 2, we have proved that a malicious user cannot obtain additional keys regarding the attributes he/she does not have. Hence, malicious users cannot deceive **DSP** to obtains ciphertext via impersonation attack.

Lemma 5: Malicious users cannot collude to obtain the data owner's ciphertext, addressing Threat 5.

    Proof: On the one hand, users from different sharing instances cannot combine their keys, due to the use of different *GID*s. On the other hand, the real attribute set $[u]$ is recorded in smart contract when authorities issue the corresponding encrypted keys. Then, the execution of judgeAttrs($[u], acp$) guarantees the authenticity of each unique user, eliminating the possibility of key combination from different users. Therefore, users cannot collude to obtain data owner's ciphertext.

D. Incentive Mechanism

We have shown how to check the honesty of an attribute authority using checkKey in Section IV-B. However, an authority may behave slackly or just abort the system due to no benefits. In this section, we introduce an incentive mechanism to enhance the participation of authorities. In our system, data owners and attribute authorities would gain benefits from data users, due to sharing data and issuing decryption keys, respectively. Furthermore, fairness is guaranteed when owners share data and users make deposits.

We introduce an owner-driven approach, in which data owners set a reasonable benefit for attribute authorities when sharing data. The pseudocodes in Algorithm 2 describe the operations each entity performs in the incentive smart contract of the data sharing paradigm. Expect is the function to input an expected profit for a data owner. Deposit is the function to apply for a mortgage for a potential data user. Withdraw is the function to withdraw deposited digital currency for data users. Reward is the function invoked automatically by the smart contract if data sharing is accomplished between data users and data owners.

In the pseudocodes, msg is an embedded variable, where $msg.sender$ represents a transaction invoker; $msg.value$ is the transferred value measured in Ether. $pool$ and $expects$ are self-defined global variables to store deposited values and expected profits. $addr_u$, $addr_o$ and $Addrs$ are the addresses of the data user, data owner and honest authorities, respectively.

Also, we can modify the incentive smart contract into a user-driven approach, in which data users are the initiators to put forward a deposit proposal. We omit the details on the modification due to space limitations.

E. Our Construction

Suppose an owner shares its data with users, leveraging the RW decentralized CP-ABE to protect the privacy of the data. We assume the RW CP-ABE *GlobleSetup* is initialized in a decentralized way. Fig. 3 shows the sequence

---

Algorithm 2 Incentive contract

  ▷ Global variables
global $pool \leftarrow \{\}$
global $expects \leftarrow \{\}$
function Expect(*GID*, ownerVal):% by a data owner
  1: $expects[GID] = $ ownerVal
  2: **return** True
function Deposit(*GID*):% by a data user
  1: $pool[msg.sender][GID] = msg.value$
  2: **return** True
function Withdraw(*GID*):% by a data owner
  1: assert($pool[msg.sender][GID]>0$)
  2: $msg.sender$.transfer($pool[msg.sender][GID]$)
  3: $pool[msg.sender][GID] = 0$
  4: **return** True
function Reward($addr_u, addr_o, Addrs, GID$): % by anyone
  1: assert($pool[addr_u][GID] > expects[GID]$)
  2: ownerVal=$expects[GID]$
  3: $addr_o$.transfer(ownerVal)
  4: avg=(deposits$[addr_u][GID]$-ownerVal)$/Addrs$.length
  5: **while** $addr_{aa} \in Addrs$ **do**
  6:     $addr_{aa}$.transfer(avg)
  7: **end while**
  8: deposits$[addr_u][GID] = 0$
  9: **return** True

---

diagram of the process, which is divided into five phases: **setup**, **encrypt**, **request**, **verify** and **access**. In our construction, we use *GID* as the unique identifier for each data-sharing process.

1) **setup**. Each authority invokes ABEAuthSetup to obtain his/her key pair (pk_θ, sk_θ) and a data user generates his/her key pair ($y, pk_u = g_1^y$). Then, public keys are published on a blockchain so that they are shared. Finally, the data owner collects authorities' public keys $\{pk_\theta\}$.

2) **encrypt**. The data owner encrypts a random value M to get ciphertext $C \leftarrow $ ABEEncrypt($M, acp, GP, \{pk_\theta\}$), where acp is a well-designed access control policy. Further, the owner encrypts the data m as $m \oplus$ kdf(M), where kdf is a key derivation function and $\oplus$ means exclusive-or operation. Then, the owner stores (ct, C) on DSP. Simultaneously, the data owner sets an expected profit value v by invoking Expect(*GID*, v).

3) **request**. The data user sends a request, containing a set of tuples ($GID, \{u, pk_\theta\}_{\theta\ in\ acp}$), to the blockchain after invoking Deposit(*GID*) to deposit some digital currency. Then, the blockchain notifies authorities that appeared in acp. Authority θ fetches the tuple (GID, pk_u, u), where pk_u is the user's public key. Next, the authority θ invokes ABEEncKey($GID, GP, u, sk_\theta, pk_u$) to produce an encrypted CP-ABE key (EK_0, EK_1). Simultaneously, the corresponding NIZK proofs are generated using algorithm genProofs. Finally, the authority θ sends (EK_0, EK_1, proofs) back to the blockchain.
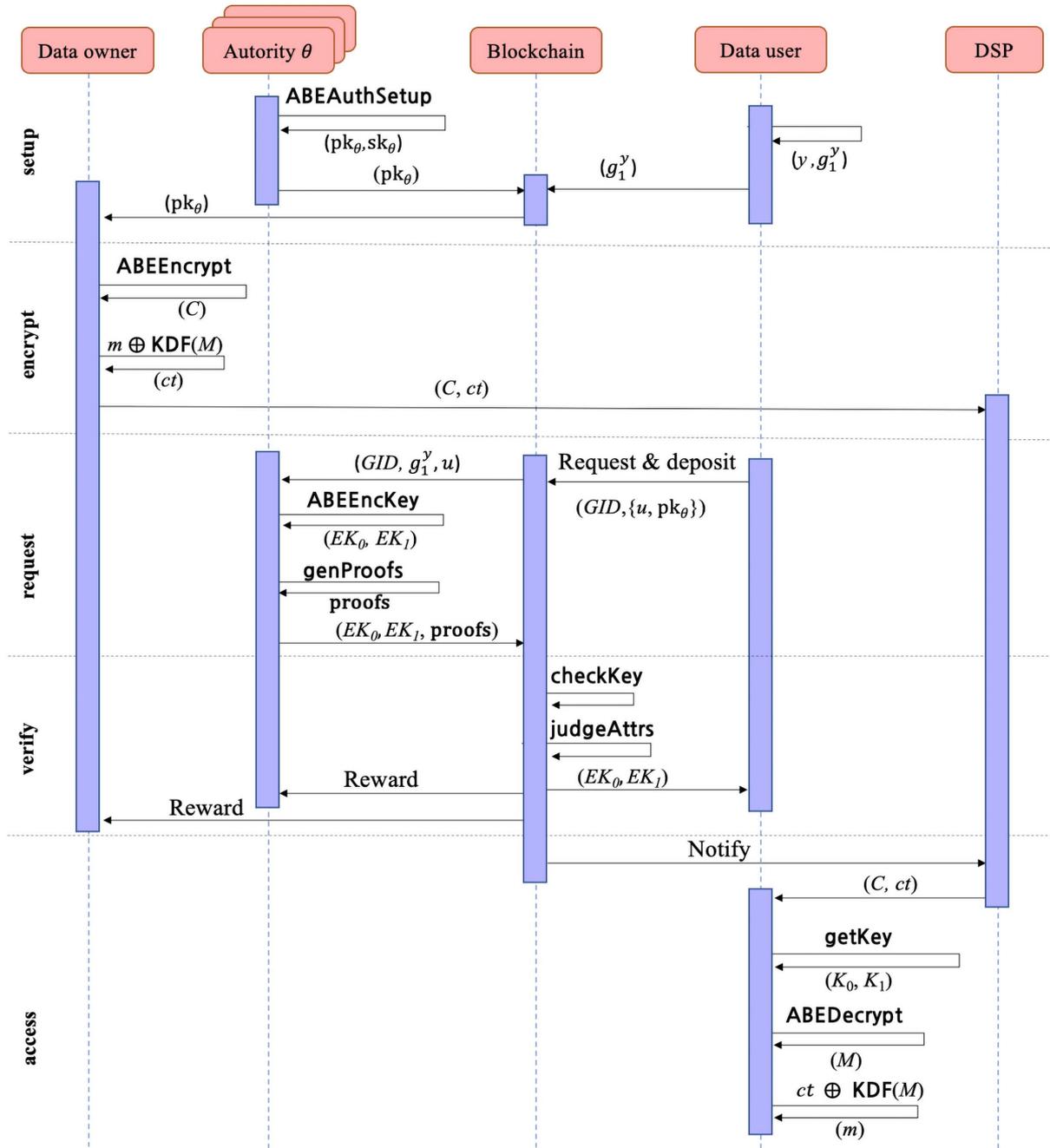
Fig. 3. Sequence diagram of the blockchain-driven data sharing paradigm for the metaverse based on CP-ABE.

4) **verify**. Upon receiving the tuple $(EK_0, EK_1, \text{proofs})$ from authority θ, the blockchain smart contract invokes $\text{checkKey}(EK_0, EK_1, \text{proofs}, GID, u)$ algorithm automatically. If checkKey outputs true, meaning the authority θ is honest, the blockchain notifies the data user to fetch (EK_0, EK_1). To eliminate the off-chain colluding attack by the authorities, the smart contract invokes $\text{judgeAttrs}([u], acp)$ to judge whether acp in the ciphertext C is satisfied by enough CP-ABE keys $\{EK_0, EK_1\}$. Once satisfied, the deposited digital currency is automatically transferred to each authority and

the data owner as incentives using $\text{Reward}(addr_u, addr_o, Addrs, GID)$.

5) **access**. Furthermore, the smart contract notifies **DSP** that the data user is qualified to access the ciphertext (C, ct). After obtaining the ciphertext (C, ct), the data user calculates RW CP-ABE decryption key (K_0, K_1) by invoking $\text{getKey}(EK_0, EK_1, g_1^{\alpha\theta}, y)$. Next, the data user invokes $\text{ABEDecrypt}(GP, C, \{K_0, K_1\})$ algorithm to obtain the value M. Finally, the original data m shared by the data user is accessed as $ct \oplus \text{kdf}(M)$.

TABLE I
COMPARISON OF GROUP COMPUTATION COST (MS)

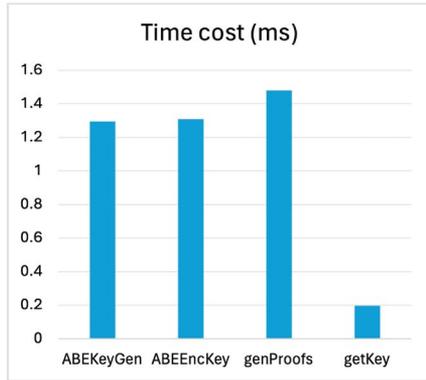| Language	Multiply$_{\mathbb{G}_1}$	Multiply$_{\mathbb{G}_2}$	Multiply$_{\mathbb{G}_T}$	Pairing
Python	10	40	190	700
Golang	0.1	0.3	0.7	2

Fig. 4.   Cost of offchain operations (ms).

Remark: If not enough authorities issue decryption keys for a data user, the user can invoke Withdraw(GID) to withdraw its deposited Ether.

V. EVALUATION

A. Experiments

We simulate the proposed system on Ubuntu 22.04-VMware workstation, powered by an Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz and 4GB of RAM. Ganache 2.7.0 is leveraged as the Ethereum test network. The embedded elliptic curve on Ethereum is "bn128" [32] and it only supports pairing-check [33], rather than bilinear mapping calculation. The onchain codes are written using Solidity language and the offchain codes are developed using Golang. The reason to choose Golang as offchain language lies in Golang is the programming language of EVM. Moreover, Golang has high performance in elliptic curve operations. Table I has shown that performance of Golang is hundreds of times better than Python, benchmarking based on Ethereum official "bn128" library. The proof-of-concept implementation of the proposed system is published on GitHub.[2]

In the proposed framework, data owners use ABEEncrypt to protect the privacy of their data; authorities invoke ABEEncKey to generate CP-ABE decryption keys; blockchain audit the authorities' honesty by invoking checkKey and judgeAttrs; data users apply getKey and ABEDecrypt to get the shared data of data owners.

We have replace the ABEKeyGen with ABEEncKey to protect the privacy of CP-ABE decryption keys. Fig. 4 shows the time cost of off-chain operations for each attribute. It shows that both ABEKeyGen and ABEEncKey algorithms cost about
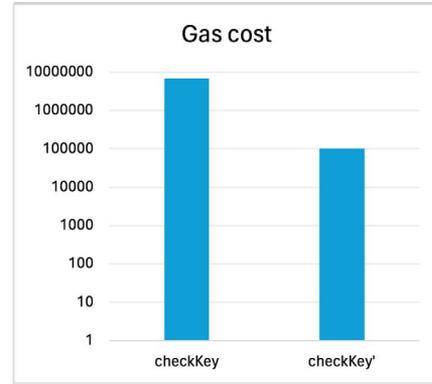
Fig. 5.   Gas cost of single checkKey and checkKey′.

1.3ms on curve bn128. The corresponding NIZK proofs generated by the genProofs algorithm cost about 1.4ms. The getKey algorithm, which calculates the original decryption key, costs about 0.19ms.

Due to high cost of $\mathbb{G}_2$ operation in Ethereum[3], checkKey defined in Section IV-B is slightly modified. We use native-supported $\mathbb{G}_1$ operation and apply pairing-check to replace the $\mathbb{G}_2$ operation to lower the gas consumption.

checkKey′$(EK_0, EK_1, \text{proofs}, GID, u):$

$$(pk_u)^{w_1} H(GID)^{w_2} F(u)^{w_3} \stackrel{?}{=} EK_0' \cdot EK_0^c$$
$$g_1^{w_3} \stackrel{?}{=} EK_2' \cdot EK_2^c$$
$$e(EK_2, g_2) \stackrel{?}{=} e(g_1, EK_1)$$
$$e(pk_u, g_2^{\alpha_\theta})e(H(GID), g_2^{\beta_\theta})e(F(u), EK_1) \stackrel{?}{=} e(EK_0, g_2)$$

where $EK_2 = g_1^{d_\theta}, EK_2' = g_1^{d_\theta'}$ and they are appended into NIZK proofs.

Fig. 5 shows the gas cost of judging whether a decryption key is valid. The gas consumption for each attribute using checkKey and checkKey′ are about 6.8×10^6 and 0.1×10^6, respectively. Apparently, checkKey′ is about much more gas-efficient than checkKey, as predicted.

Smart contract uses judgeAttrs to check whether a combination of keys comprises a valid decryption key for specific CP-ABE ciphertexts. By randomly choosing AND and OR gates in the access control policy, we evaluate the gas cost of judgeAttrs and checkKey′ for all attributes in Fig. 6. The total validation cost, as **verify** needs, can be calculated by directly adding the gas cost of judgeAttrs and checkKey′.

Fig. 7 depicts the gas cost of functionalities in the incentive smart contract. It can be indicated that the gas consumption for Expect, Deposit and Withdraw is constant. The gas consumption are fixed at 52248, 29182 and 47035, respectively. The gas cost of Reward functionality increases linearly with the number of honest authorities involved in a data sharing process.

Fig. 8 demonstrates the CP-ABE ciphertext size and (encrypted) key size. It can be seen that the ciphertext size is about 147KB when 100 attributes appeared in acp and the
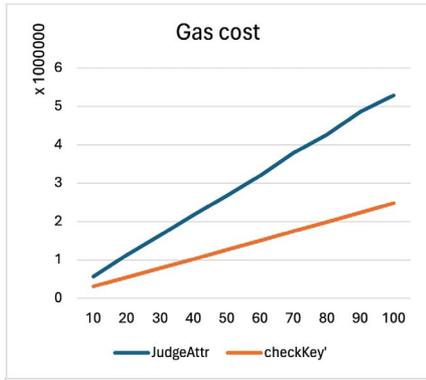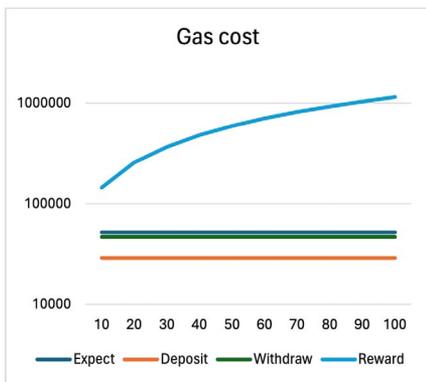
Fig. 6. Total gas cost of judgeAttrs and checkKey′.
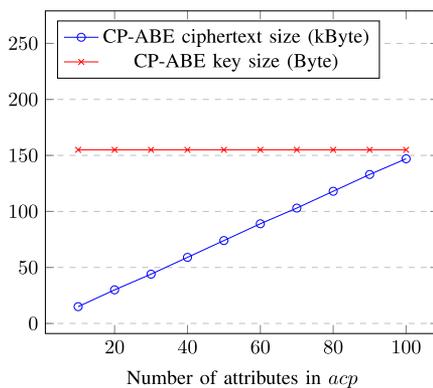

Fig. 7. Gas cost of incentive smart contract.

Fig. 8. Cipheretxt size (KByte) and key size (Byte).

Fig. 9. Encryption cost (ms).

Fig. 10. Decryption cost (ms).

key size is fixed at 155B. Each attribute authority costs a constant time, i.e., 1.23ms, to generate a key for each attribute string. We have combined CP-ABE with symmetric encryption, i.e., leveraging CP-ABE plaintext as symmetric key seed. Since KDF is employed to generate symmetric key and XOR is leveraged as encryption/decryption operation. Further, we evaluate the performance of these offchain operations using Golang implementation. Figs. 9 and 10 depict the total encryption 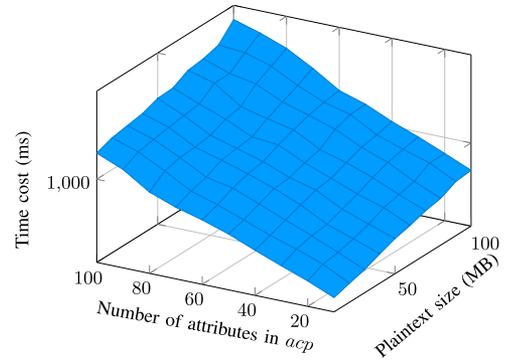and d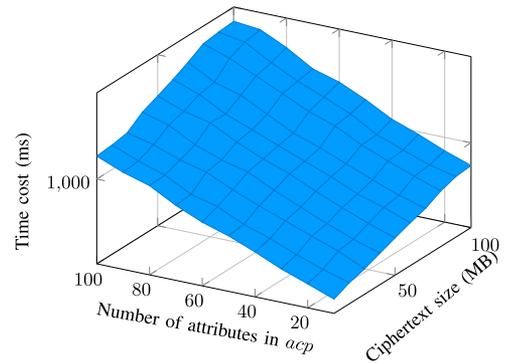ecryption cost considering both attributes number and plaintext/ciphertext size, respectively. With 100 attributes in acp and 100MB plaintext/ciphertext, the total time overhead of encryption (or decryption) is about 1.8s.

Denote E as an exponentiation on a group and P as a bilinear pairing. From Table I, we can know that pairing is much more costing than exponentiation. Han et al. [24], Guo et al. [14] and Yang et al. [20] are some of the most recent decentralized CP-ABE with accountability. Table II compares the performance regarding CP-ABE key generation, the associating NIZK proof generation and key accountability/verification. It can be seen that only our protocol protects key privacy and other works [14], [20], [24] expose keys when considering key accountability. As a sequence, we encounter additional overhead to generate the NIZK proofs, which is verified by the checkKey algorithm. In checkKey, pk_u, $g_2^{\alpha_\theta}$, $H(GID)$ and $g_2^{\beta_\theta}$ are public available. Hence, only $2n + 2$ bilinear pairings is required, where n is the number of attributes appeared in a key from a single authority. Fig. 11 indicates that our protocol requires lower computational overhead in key verification (or accountability) than other works [14], [20], [24].

B. Property Analysis

The proposed scheme satisfies the system goals defined in Section III-C and we make a analysis as follows.
1) *Tamper-resistance*: Tamper-resistance of the proposed scheme directly inherits from the property of blockchain.

TABLE II
COMPARISON REGARDING KEY ACCOUNTABILITY. THE NOTATIONS n, P AND E DENOTE THE NUMBER OF ATTRIBUTES, BILINEAR MAPPING COST AND EXPONENTIATION COST, RESPECTIVELY

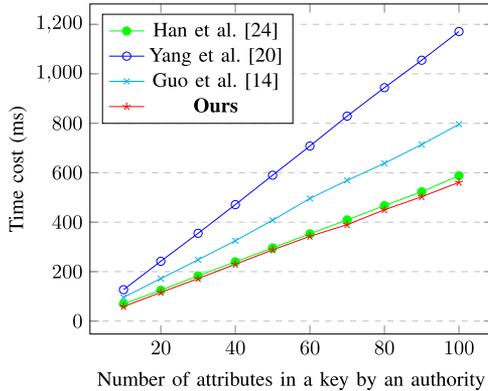| Reference	Key Protection	Key Generation	Proof Generation	Key size	Proof size	Verification
Han et al. [24]	✗	$(2n+5)E$	-	$(n+4)\|\mathbb{G}\| + (logn+1)\|\mathbb{Z}_p\|$	-	$(3n+4)P + (n+2)E$
Yang et al. [20]	✗	$5nE$	-	$3n\|\mathbb{G}\|+ 2\|\mathbb{Z}_p\|$	-	$5nP + 4nE$
Guo et al. [14]	✗	$(4n+4)E$	-	$(2n+3)\|\mathbb{G}\|+ \|\mathbb{Z}_p\|$	-	$(4n+4)P + (2n+2)E$
Ours	✓	$(2n+2)E$	$4nE$	$2n\|\mathbb{G}\|$	$2n\|\mathbb{G}\|+ (3n+1)\|\mathbb{Z}_p\|$	$(2n+2)P + 6nE$

Fig. 11. The computational overhead regarding key verification or accountability.

With Ethereum blockchain storing encrypted CP-ABE keys, keys are untamperable. Also, **DSP** is a buisiness-related trusted service. Thus, encrypted messages or files are unalterable.

2) *Decentralization*: On the one hand, Ethereum blockchain is a fully decentralized system, without a privileged administrator or manager. On the other hand, decentralized CP-ABE is leveraged as the underlying cryptographic primitive, allowing multiple attribute authorities to make a decision collectively in data sharing.

3) *Key privacy*: We slightly modify the key generation algorithm so that the output protects privacy of a valid CP-ABE key, as ABEEncKey shows. We also demonstrate that the algorithm is resistant to eavesdropping attacks by Lemma 1. The targeted user can recover the legal decryption key with its private key, as getKey demonstrates. Besides, we argue that the malleability key described in Lemma 2 cannot generate decryption keys of new attributes. Thus, the output of ABEEncKey can be delivered publicly through blockchain without leaking key privacy.

4) *Data privacy*: **DSP** provides available data storage to authorized users. To resist honest but curious **DSP**, data user encrypts its data by applying RW CP-ABE before uploading the data. The underlying RW CP-ABE is secure based on the q-DPBDHE2 assumption [11]. We have proved that the public output of ABEEncKey does not reveal information about (K_0, K_1). We leverage judgeAttrs to judge whether a user is authorized to access ciphertext stored on **DSP**. The judgment is accomplished publicly

on Ethereum smart contract. Lemma 4 and Lemma 5 show that owner's ciphertext and digital asset are private against malicious users. Thus, only qualified users (with legal keys and enough deposits) can access the ciphertext, further eliminating the risks of data leakage.

5) *Forward-secrecy*: *GID* enables the decentralized CP-ABE scheme to resist collusion attack [11], [12], making CP-ABE keys unaggregatable given different *GID*. In our design, each data-sharing instance corresponds to a unique *GID*, making different sharing instances separately. Thus, a decryption key is only valid for a unique ciphertext. Moreover, even if a malicious user gets a valid decryption key, it cannot access the ciphertext from **DSP**, as clarified by Lemma 4. Hence, when a decryption key is leaked, the privacy of previously shared data is still guaranteed.

6) *Incentives*: In our protocol, we design an owner-driven approach — data owner sets the expected benefit when sharing its data using Expect on smart contract. Subtracting the required digital currency of data owner, the rest of deposited currency is transferred to honest authorities as rewards. Hence, both data owner and honest authorities will gain incentives in the paradigm.

7) *Accountability*: The proposed framework require authorities to issue CP-ABE keys using the ABEEncKey algorithm. Additionally, NIZK proofs are attached for the encrypted CP-ABE keys, as demonstrated by the genProofs algorithm. Lemma 3 proves that authorities are accountable with the NIZK proofs. Moreover, the accountability is automatically accomplished on smart contract.

8) *Key abuse resistance*: We first consider the illegal authorization problem. The *Accountability* property guarantees that authorities cannot issue invalid keys. Besides, the process of key issuance (i.e., authorization) to each target user is transparent and publicly verifiable. Then, we discuss the key disclosure problem. Lemma 1 and Lemma 2 highlight that malicious users cannot construct meaningful keys. Lemma 5 implies that exposing key to another user is useless in obtaining data owner's digital assets. Hence, key disclosure problem is resolved.

VI. RELATED WORKS

A. Accountability in CP-ABE

Liu et al. [22] propose to reduce trust in authorities by introducing a central authority (CA) and tracing authority (TA)

in a white-box traceable CP-ABE scheme. Later, Yu et al. [16] put the user's identity into the CP-ABE decryption key. Thus, any third party can recognize the user directly from the exposed decryption key. They use a digital signature to make authorities publicly verifiable. To uncover the misbehavior of malicious authorities, they put forward two kinds of authorities: namely, central authority (CA) and outsourced authority (OA). These two authorities are assumed not to collude with each other.

Ning et al. [15] define an accountable authority for the first time. If a malicious authority distributes legitimate keys to unauthorized users, such behaviors can be discovered. The main idea is to make the data user's decryption key jointly determined by the authority and the user. Later, Li et al. [17] compose a CP-ABE scheme achieving authority accountability in their proposed hidden-policy ABE scheme. More recently, Liu et al. [41] put forward a black-box accountable CP-ABE scheme to identify the owner of the faked decryption device and the malicious authority. Hei et al. [13] employ a dual receiver cryptosystem (DRC) [34] to generate key pairs for users and the key auditor. In the proposed CP-ABE algorithm, the authorities generate and encrypt keys using the corresponding user's public key. Guo et al. [14] uses CP-ABE to protect privacy in VANET data sharing, where a consortium blockchain is incorporated to resist key abuse of AAs. For user key abuse, they achieve white-box traceability, as previous researches [15], [17], [22] do.

B. Key Abuse in CP-ABE

Chase et al. [35] introduce an anonymous key issuing protocol based on a two-party secure computation (2PC) protocol to enhance user privacy. Qiao et al. [36] regard the key disclosure problem as privilege abuse. Then, they present a black-box compulsory traceable CP-ABE scheme by introducing a tracing ciphertext. The scheme guarantees that the adversary cannot distinguish the tracing ciphertext from the normal ciphertext, thus addressing privilege abuse. Cui et al. [37] define the concept of key regeneration-free for CP-ABE, aiming to eliminate key abuse by illegal data users. The key idea is to introduce randomness in different parts of the decryption key. Later, Zhang et al. [38] apply the key regeneration-free technique to achieve strong key unforgeability in a lightweight data sharing scheme. Hei et al. [13] design a key audit scheme using an incentive mechanism to address the key leakage issue. To protect the privacy of attribute keys, they incorporate dual receiver cryptosystem techniques. Meanwhile, the illegal authorization issue is addressed through transparent storage on the blockchain. However, this mechanism relies on a trusted role (auditor), who is used to detect malicious behaviour.

Recently, Liu et al. [39] conduct comprehensive key management in CP-ABE for cloud data with a consortium blockchain. Specifically, they leverage Hyperledger Fabric [40], where distributed nodes collectively keep secret keys via Pedersen's distributed key generation (DKG) technique. When using DKG, they leverage blockchain to achieve consensus among validators. They further divide decryption phase into two parts. Partial decryption uses a publicly known decryption key from the blockchain and full decryption makes use of a user secret key.

Consequently, their approach achieves key escrow-free and key distribution-free. Moreover, they are able to trace the illegal user identity without exposing the CP-ABE decryption key. However, they do not provide the feature of authority accountability.

C. Data Sharing Using CP-ABE

Ren et al. [21] introduce a blockchain-based CP-ABE, where ciphertext and keys are distributed and managed in blockchain. They incorporate a threshold proxy re-encryption to protect data owner's master key. They also consider a staking economic incentive model with reward or slashing to ensure nodes security. Qin et al. [42] put forward a blockchain-based multi-authority access control scheme for sharing data securely. They exploit smart contracts to compute tokens for attributes managed across multiple management domains. Moreover, blockchain is applied to record the access process so that the access history can be audited.

Wang et al. [18] develops a privacy-preserving medical data sharing system by using blockchain, thereby eliminating the single-point-of-failure problem. They design a non-interactive on-chain search protocol to enable multiple users to search and access electronic health records. Murad et al. [26] propose a P2P architecture within fog nodes to handle client requests, thereby reducing the cloud overhead and enhancing efficiency. They leverage a CP-ABE scheme to obtain a fine-grained and secure communication channel. Mahdavi et al. [27] downgrade standard ABE to fuzzy identity-based encryption to lower computation in IoT devices. Further, they outsource exponentiation and bilinear pairing of CP-ABE scheme to cloud servers. Zhang et al. [28] design a revocable CP-ABE scheme to resist economic denial of sustainability attack in data sharing, where blockchain is employed to judge the download permission of data users. Yang et al. [20] propose a traceable privacy-preserving data sharing for fog-based smart logistics based on multi-authority CP-ABE.

Zhang et al. [25] protect the privacy of Metaverse healthcare data by presenting a CP-ABE scheme with constant computation overhead, which applies a multi-server architecture. To minimize storage space caused by invalid and duplicated ciphertexts, they introduce a mechanism to check the validity and equivalence of ciphertexts. Additionally, they introduce an attribute-based re-encryption technique to enable authority delegation after deduplication.

Other works [46], [47], [48], [49] also address data sharing using blockchain and CP-ABE, focusing on various aspects such as attribute revocation, ciphertext access strategies, update capabilities, and invisibility.

D. Comparison With Related Works

Blockchain is a promising infrastructure for sharing data among distributed enterprises or entities. Sifah et al. [19] propose a method to selectively share outsourced encrypted data in the cloud by embedding an authorization policy to data. They claim to achieve data traceability or auditability, because blockchain is employed to record tamper-proof logs and activities.

TABLE III
COMPARISON WITH RELATED WORKS

Approach	Data sharing	Decentralization	Accountability	Forward-secrecy	Key-privacy	Incentives	Key abuse resistance
Sifah et al. [19]	✓	✓	✓	✗	✗	✗	✗
Li et al. [17]	✓	✗	✓	✗	✗	✗	✓
Chase et al. [35]	✗	✓	✗	✓	✗	✗	✓
Liu et al. [39]	✓	✓	✗	✓	✗	✗	✓
Zhang et al. [25]	✓	✗	✓	✗	✗	✗	✗
Guo et al. [14]	✓	✓	✓	✓	✗	✗	✓
Hei et al. [13]	✓	✓	✓	✗	✗	✓	✓
Ren et al. [21]	✓	✓	✓	✗	✓	✓	✗
Ours	✓	✓	✓	✓	✓	✓	✓

Liu et al. [39] and Guo et al. [14] resolve the key leakage problem by invalidating the user's key and invoking a key update algorithm, respectively. Some systems [14], [35], [39] have considered the single point failure problem, hence achieving decentralization. Zhang et al. [25] mainly focus on CP-ABE ciphertext management, rather than keys, in metaverse healthcare. Ren et al. [21] manage CP-ABE keys with a threshold proxy re-encryption scheme.

Table III summarizes the comparison with related works regarding the achieved features. Some works [17], [25] are not applicable to the metaverse environment due to the use of single authority CP-ABE. The 2PC protocol which enables user anonymity in Chase et al. [35] is interactive and requires a secure communication channel, leading to obstacles in application in metaverse. Sifah et al. [19] focus on access control policy which enables to outsource data selectively and omit the discussion of key generation and delivery, thus inapplicable for metaverse. Other works [13], [14], [17], [39] implement traceability of CP-ABE keys, and the "trace" methods input CP-ABE keys. That means decryption keys should be presented for accountability, which is impractical in a public metaverse environment. The sharing scheme provided by Ren et al. [21] can be applied in metaverse with a different system model, since additional proxies should be considered.

VII. APPLICATION IN GAMEFI

GameFi, allowing players to earn real-world rewards through gameplay, serves as a financial layer that enriches the interactive and economic aspects of the metaverse. GameFi often incorporates elements like play-to-earn (P2E) mechanics, governance tokens, and decentralized autonomous organizations (DAOs), enabling players not only to earn income but also to participate in decision-making processes that shape the game's development and broader ecosystem. By merging gaming and financial incentives, GameFi aims to foster a more engaging and community-driven metaverse.

If the game provider (i.e., **DSP** in this context) establishes all the rules, it may reduce player engagement and trust in the game. Such centralized control can lead players to feel a lack of autonomy and fairness, which can diminish their motivation to participate. To enhance the game's appeal, providers should

consider adopting more open and transparent rules, encouraging player involvement in the decision-making process, and fostering a greater sense of belonging and investment among players. However, ensuring the security and privacy of user data and digital assets is paramount in GameFi.

The proposed decentralized CP-ABE based framework with accountable authority offers a compelling solution to address these challenges. With the framework, players can stake their money in smart contract to become CP-ABE authorities and they can interact with game provider with special APIs. Then, each user attribute (e.g., age, level, role, achievement, region, sex, playtime, skill, membership, balance, etc.) is accessible only to specific authorities. Essentially, these authorities form a decentralized autonomous organization (DAO) [43] with accountability. Game players or providers can leverage the CP-ABE encryption algorithm to secure various virtual assets, including skins, weapons, and other in-game items. By defining access policies that correspond to user attributes, players or providers can ensure that only those who meet the criteria can access or trade these assets. This not only enhances security but also creates a more engaging gaming experience, as players are rewarded based on their accomplishments.

Suppose each in-game item is bound to an NFT [44] address. When a player (or the game provider) wishes to trade an in-game item, he/she encrypts the description, which demonstrates the NFT ownership, with a proper access control policy. For example, a skin of player1 in an online game is encrypted with an access control policy acp=((level$\geq$25@AUTH1 OR cityLA@AUTH2) AND female@AUTH3). Then, player2 with attribute $[u_1]$=[level25@AUTH1, cityPHX@AUTH2, female@AUTH3] can request to authorities for decryption keys to obtain the ownership of the skin. However, player3 with attributes $[u_2]$=[level28@AUTH1, cityLA@AUTH2, male@AUTH3] is unable to buy the item. AUTH1, AUTH2 and AUTH3 are different authority identities, demonstrating that each authority has separated attribute spaces. As described by the framework, the encrypted keys sent to player2 are generated using **ABEEncKey** algorithm, along with **genProofs** being invoked. The encrypted keys and proofs are uploaded to smart contract. Finally, player2 can run **getKey** to obtain valid keys and execute the CP-ABE decryption algorithm to obtain player1's on-sale skin. After a successful

trade of an in-game item, the authorities gain rewards for providing key issuance services, as the Reward algorithm shows. In case an authority is caught misbehaving in the service which can be detected by the checkKey algorithm, its stake will be forfeited by the smart contract.

Similarly, the game provider, acting as a data seller, can issue new in-game merchandise through this framework. This approach minimizes the risks associated with the arbitrary issuance of merchandise by the game provider, enhancing transparency and credibility, as the entire inventory of merchandise is publicly accessible.

Clearly, the proposed framework will undoubtedly increase players' enthusiasm for participating in the game, thereby enhancing the overall prosperity of the game ecosystem.

VIII. CONCLUSION AND FUTURE WORK

This paper proposes a blockchain-driven data sharing paradigm for the metaverse based on the RW decentralized CP-ABE scheme. The data sharing paradigm achieves secure key management, where the key abuse problem is addressed and authority accountability is considered. We leverage blockchain to store encrypted CP-ABE keys, to which NIZK proofs are attached. The keys are verified publicly in smart contract after authorities publish them. Besides, data users are qualified to obtain the keys only if they deposit enough digital currency in another smart contract. Data owners and authorities are rewarded after users access data successfully. Hence, the proposed paradigm gets rid of the key disclosure and illegal authorization problems. Compared with related works, we are the first to realize authority accountability with key privacy.

We plan to further refine the application of the proposed framework in GameFi, such as enabling DAO governance and implementing in-game item rentals and exchange. Additionally, blockchains and smart contracts inherently lack the ability to access data outside their network (i.e., off-chain data). In the future, we plan to build a decentralized oracle based on the proposed data sharing paradigm, making full use of accountability of CP-ABE authorities and fine-grained access control. The oracle aims at providing services that manage external data outside blockchain in a trustworthy and decentralized manner.

REFERENCES

[1] W. Gan, Z. Ye, S. Wan, and P. Yu, "Web 3.0: The future of internet," 2023, *arXiv:2304.06032*.

[2] X. Dong, J. Yu, Y. Luo, Y. Chen, G. Xue, and M. Li, "Achieving an effective, scalable and privacy-preserving data sharing service in cloud computing," *Comput. Secur.*, vol. 42, pp. 151–164, 2014.

[3] H. Xiong, H. Zhang, and J. Sun, "Attribute-based privacy-preserving data sharing for dynamic groups in cloud computing," *IEEE Syst. J.*, vol. 13, no. 3, pp. 2739–2750, Sep. 2019.

[4] S. Wang, K. Liang, J. Liu, J. Chen, J. Yu, and W. Xie, "Attribute-based data sharing scheme revisited in cloud computing," *IEEE Trans. Inf. Forensics Secur.*, vol. 11, no. 8, pp. 1661–1673, Aug. 2016.

[5] L. Zhang, H. Kan, and H. Huang, "Patient-centered cross-enterprise document sharing and dynamic consent framework using consortium blockchain and ciphertext-policy attribute-based encryption," in *Proc. CF*, 2022, pp. 58–66.

[6] K. Yu, L. Tan, M. Aloqaily, H. Yang, and Y. Jararweh, "Blockchain-enhanced data sharing with traceable and direct revocation in IIoT," *IEEE Trans. Ind. Inform.*, vol. 17, no. 11, pp. 7669–7678, Nov. 2021.

[7] C. Feng et al., "Efficient and secure data sharing for 5G flying drones: A blockchain-enabled approach," *IEEE Netw.*, vol. 35, no. 1, pp. 130–137, Jan./Feb. 2021.

[8] Y. Wang et al., "SPDS: A secure and auditable private data sharing scheme for smart grid based on blockchain," *IEEE Trans. Ind. Inform.*, vol. 17, no. 11, pp. 7688–7699, Nov. 2021.

[9] L. Zhang, F. Qiu, F. Hao, and H. Kan, "1-round distributed key generation with efficient reconstruction using decentralized CP-ABE," *IEEE Trans. Inf. Forensics Secur.*, vol. 17, pp. 894–907, 2022.

[10] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Secur. Privacy (SP'07)*, 2007, pp. 321–334.

[11] Y. Rouselakis and B. Waters, "Effcient statically-secure large-universe multi-authority attribute-based encryption," in *Proc. FC*, 2015, pp. 315–332.

[12] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Proc. Eurocrypt*, 2011, pp. 568–588.

[13] Y. Hei, J. Liu, H. Feng, D. Li, Y. Liu, and Q. Wu, "Making MA-ABE fully accountable: A blockchain-based approach for secure digital right management," *Comput. Netw.*, vol. 191, no. 108029, 2021.

[14] Z. Guo, G. Wang, Y. Li, J. Ni, R. Du, and M. Wang, "Accountable attribute-based data sharing scheme based on blockchain for vehicular ad hoc network," *IEEE Internet Things J.*, vol. 10, no. 8, pp. 7011–7026, Apr. 2023.

[15] J. Ning, X. Dong, Z. Cao, and L. Wei, "Accountable authority ciphertext-policy attribute-based encryption with white-box traceability and public auditing in the cloud," in *Proc. ESORICS*, 2015, pp. 270–289.

[16] G. Yu, X. Ma, Z. Cao, G. Zeng, and W. Han, "Accountable CP-ABE with public verifiability: How to effectively protect the outsourced data in cloud," *Int. J. Found. Comput. Sci.*, vol. 28, no. 6, pp. 705–723, 2017.

[17] J. Li, Y. Zhang, J. Ning, X. Huang, G. Poh, and D. Wang, "Attribute based encryption with privacy protection and accountability for CloudIoT," *IEEE Trans. Cloud Comput.*, vol. 10, no. 2, pp. 762–773, Apr./Jun. 2020.

[18] M. Wang, Y. Guo, C. Zhang, C. Wang, H. Huang, and X. Jia, "MedShare: A privacy-preserving medical data sharing system by using blockchain," *IEEE Trans. Services Comput.*, vol. 16, no. 1, pp. 438–451, Jan./Feb. 2023.

[19] E. Sifah, Q. Xia, H. Xia, K., K. Agyekum, C. Cobblah, and J. Gao, "Selective sharing of outsourced encrypted data in cloud environments," *IEEE Internet Things J.*, vol. 8, no. 18, pp. 14141–14155, 2021.

[20] Y. Yang, J. Zhang, X. Liu, and J. Ma, "A scalable and auditable secure data sharing scheme with traceability for fog-based smart logistics," *IEEE Internet Things J.*, vol. 10, no. 10, pp. 8603–8617, May 2023.

[21] Z. Ren, E. Yan, T. Chen, and Y. Yu, "Blockchain-based CP-ABE data sharing and privacy-preserving scheme using distributed KMS and zero-knowledge proof," *J. King Saud Univ. Comput. Inf. Sci.*, vol. 36, no. 3, Mar. 2024, Art. no. 101969.

[22] Z. Liu, Z. Cao, and D. Wong, "White-box traceable ciphertext-policy attribute-based encryption supporting any monotone access structures," *IEEE Trans. Inf. Forensics Secur.*, vol. 8, no. 1, pp. 76–88, Jan. 2013.

[23] C. Yuan, M. Xu, X. Si, and B. Li, "Blockchain with accountable CP-ABE: How to effectively protect the electronic documents," in *Proc. ICPADS*, 2017, pp. 800–803.

[24] D. Han, N. Pan, and K. Li, "A traceable and revocable ciphertext-policy attribute-based encryption scheme based on privacy protection," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 1, pp. 316–327, Jan./Feb. 2022.

[25] T. Zhang, J. Shen, C. Lai, S. Ji, and Y. Ren, "Multi-server assisted data sharing supporting secure deduplication for metaverse healthcare systems," *Future Gener. Comput. Syst.*, vol. 140, pp. 299–310, Mar. 2023.

[26] S. Murad and N. Rahimi, "Secure and efficient hierarchical P2P fog architecture: A novel approach for IoT," *IEEE Internet Things J.*, vol. 11, no. 10, pp. 18796–18807, May 2024.

[27] M. Mahdavi, M. Tadayon, M. Haghighi, and Z. Ahmadian, "IoT-friendly, pre-computed and outsourced attribute based encryption," *Future Generation Comput. Syst.*, vol. 150, pp. 115–126, Jan. 2024.

[28] Q. Zhang, C. Xu, H. Zhong, C. Gu, and J. Cui, "Revocable and efficient blockchain-based fine-grained access control against EDoS attacks in cloud storage," *IEEE Trans. Comput.*, vol. 73, no. 8, pp. 2012–2024, Aug. 2024.

[29] D. Boneh and V. Shoup, "A graduate course in applied cryptography," Draft 0.6, 2023. [Online]. Available: https://toc.cryptobook.us/book.pdf#page=769.67

[30] A. Fiat and A. Shamir, "How to prove yourself: Practical solutions to identification and signature problems," in *Proc. CRYPTO*, 1986, pp. 186–194.

[31] G. Wood, *Ethereum: A Secure Decentralised Generalised Transaction Ledger*. Ethereum Project Yellow Paper, 2014.

[32] E.I.P.-196, "Eip-196: Precompiled contracts for addition and scalar multiplication on the elliptic curve alt_bn128."

[33] E.I.P.-197, "Precompiled contracts for optimal ate pairing check on the elliptic curve alt_bn128."

[34] T. Diament, H. Lee, A. Keromytis, and M. Yung, "The dual receiver cryptosystem and its applications," in *Proc. CCS*, 2004, pp. 330–343.

[35] M. Chase and S. Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *CCS*, pp. 121–130, 2009.

[36] H. Qiao, J. Ren, Z. Wang, H. Ba, and H. Zhou, "Compulsory traceable ciphertext-policy attribute-based encryption against privilege abuse in fog computing," *Future Gener. Comput. Syst.*, vol. 88, pp. 107–116, Nov. 2018.

[37] H. Cui, R. Deng, B. Qin, and J. Weng, "Key regeneration-free ciphertext-policy attribute-based encryption and its application," *Inf. Sci.*, vol. 517, pp. 217–229, May 2020.

[38] J. Zhang, M. Wu, Q. Zhang, and C. Peng, "A lightweight data sharing scheme with resisting key abuse in mobile edge computing," in *Proc. INFOCOM*, 2021, pp. 1–6.

[39] S. Liu, J. Yu, L. Chen, and B. Chai, "Blockchain-assisted comprehensive key management in CP-ABE for cloud-stored data," *IEEE Trans. Netw. Service Manage.*, vol. 20, no. 2, pp. 1745–1758, Jun. 2023.

[40] E. Androulaki et al., "Hyperledger fabric: A distributed operating system for permissioned blockchains," in *Proc. EuroSys*, 2018, pp. 1–15.

[41] Z. Liu, Y. Ding, M. Yuan, and B. Wang, "Black-box accountable authority CP-ABE scheme for cloud-assisted e-health system," *IEEE Syst. J.*, vol. 17, no. 1, pp. 756–767, Mar. 2022.

[42] X. Qin, Y. Huang, Z. Yang, and X. Li, "A blockchain-based access control scheme with multiple attribute authorities for secure cloud data sharing," *J. Syst. Archit.*, vol. 112, no. 101854, 2021.

[43] T. Gadekallu et al., "Blockchain for the metaverse: A review," *Future Gener. Comput. Syst.*, vol. 143, pp. 401–419, 2023.

[44] W. Entriken, D. Shirley, J. Evans, and N. Sachs, "ERC721 non-fungible token standard." [Online]. Available: https://eips.ethereum.org/EIPS/eip-721

[45] T. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," in *Proc. CRYPTO*, 1991, pp. 129–140.

[46] W. Ma et al., "Attribute revocable data sharing scheme based on blockchain and CP-ABE," in *Proc. CSAE*, 2020, pp. 1–7.

[47] Z. Zhang, Z., and X. Ren, "Data security sharing method based on CP-ABE and blockchain," *J. Intell. & Fuzzy Syst.*, vol. 40, no. 2, pp. 2193–2203, 2021.

[48] J. Xue, L. Shi, W. Zhang, W. Li, X. Zhang, and Y. Zhou, "Poly-ABE: A traceable and revocable fully hidden policy CP-ABE scheme for integrated demand response in multi-energy systems," *J. Syst. Archit.*, vol. 143, no. 102982, 2023.

[49] C. Zhang, G. Zhang, and G. Liu, "Secure data sharing on the industrial internet: A decentralized scheme based on CP-ABE," in *Proc. IAI*, 2023, pp. 1–6.

Zhanrong Ou was born in 1998. He received the bachelor's degree from Hanshan Normal University, Chaozhou, Guangdong Province, in 2021. He is currently working toward the master's degree with the School of Cyberspace Security (School of Cryptology), Hainan University. He joined Hainan University in 2022. His research interests consist of blockchain and decentralized randomness beacon.

Changhui Hu received the Ph.D. degree from the School of Mathematics, Shandong University, in 2012. Currently, he is a Professor with the School of Cyberspace Security (School of Cryptology), Hainan University. His research interests include information security and cryptography. His recent publications are included in Usenix Security'21, IEEE TRANSACTIONS ON CLOUD COMPUTING, *JIS*, *CN*, and *SCN*.

Haibin Kan (Member, IEEE) was born in 1971. He received the Ph.D. degree from Fudan University, Shanghai, China, 1999. From 2002 to 2006, he was with Japan Advanced Institute of Science and Technology as an Assistant Professor. He went back Fudan University in 2006, where he is currently a Full Professor. He is also the Director of Shanghai Blockchain Engineering Research Center. His research interests include coding theory, cryptography, and computation complexity.

Liang Zhang received the bachelor's degree from Huazhong University of Science and Technology, and the Ph.D. degree from Fudan University, in 2022. He received the excellent programming skills in Baidu Company Ltd. Currently, he is an Associate Professor with Hainan University and a Postdoctoral Fellow with Hong Kong University of Science and Technology. His research interests include blockchain and applied cryptography. His recent publications are included in IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON SERVICES COMPUTING, CJ, CF'22, and SACMAT'22.

Jiheng Zhang received the B.S. degree in mathematics from Nanjing University, the M.S. degree in mathematics from The Ohio State University, and the M.S. and Ph.D. degrees in operations research from Georgia Institute of Technology. He joined Industrial Engineering and Decision Analytics (IEDA) of Hong Kong University of Science and Technology (HKUST) as an Assistant Professor in 2009, and was promoted as an Associate Professor in 2015, and a Full Professor in 2020. Currently, he is the Department Head of IEDA of HKUST. He is also the Associate Director of HKUST Crypto-Fintech Lab.