Blockchain-Based Digital Rights Management Scheme via Multiauthority Ciphertext-Policy Attribute-Based Encryption and Proxy Re-Encryption

Juntao Gao, Haiyong Yu, Xiuqin Zhu, and Xuelian Li

***Abstract*—In order to ensure the confidentiality of digital contents, improve the fairness of digital copyright transactions, and reduce the time and management overhead of digital copyright owners, we proposed a blockchain-based digital rights management scheme. First, we designed a new multiauthority ciphertext-policy attribute-based encryption (MA-CPABE) scheme and showed that the new MA-CPABE has the indistinguishability of plaintext under adaptively chosen plaintext attack (IND-CPA) security and good performance. By combining the MA-CPABE and proxy re-encryption, the rights owner can flexibly sell the copyright to different users with once encryption by an agent who cannot access any information related to digital content when changing the ciphertext access policy as required. By using the smart contract of Ethereum, a fair trade of the decryption keys between the rights owner and rights requester is implemented. In order to further improve fairness, another blockchain is used as a ledge to store information related to digital rights, which greatly reduces the storage overhead in public blockchain. Security analysis shows that our scheme can provide IND-CPA security, resist collusion attacks, and protect the user's privacy. Performance analysis shows that our scheme can provide a wealth of features to meet the various needs of users. The simulation results show that our scheme is very efficient compared to other schemes.**

***Index Terms*—Blockchain, digital rights management (DRM), multiauthority ciphertext-policy attribute-based encryption (MA-CPABE), proxy re-encryption.**

I. Introduction

THE giant increase and the new characteristics of the digital contents stimulates interest in the digital right management (DRM) tools again, which involves various access control technologies for restricting the use of proprietary hardware and copyrighted works [1]–[7]. DRM schemes are roughly divided into two types, i.e., the centralized digital rights management (CDRM) scheme and the distributed digital rights management (DDRM) scheme. In the CDRM, the user is required to delegate digital contents to a third party, and the third party is solely responsible for the digital right transfer. The advantage of the CDRM is that it provides great convenience to digital rights owners. However, the third party could violate the established rule and have malicious behavior, such as distributing the digital contents to the unauthorized users. It will definitely harm the interests of the digital rights owner. Moreover, when the server of the third party is hacked, some information related to digital content will be leaked. And if the third party cannot be accessed to, it will affect all of users. In the DDRM, the digital rights is transferred or distributed directly by digital rights owners. The advantage of the DDRM is that the digital rights owners can take control of the digital rights but this method greatly increases the time and computing overhead of digital rights owners.

The two types of DRM schemes are not very suitable for these new Internet products, such as serial stories or short videos. On the one hand, The creators of the serial stories and short videos usually do not have enough time to manage their digital content as well. Usually, they delegate their digital contents to one or several platforms for management. On the other hand, the short videos can be cut, revised and broadcast more easily by an unauthenticated party, the creators need to use more effective means to protect their rights and interests when disputes arise. Therefore, for the creators, they could be concerned about whether the platform (the third party) will modify their works and disclose their works to some unauthorized audiences without their consent. For the audience, they could concern the fairness, i.e., whether they can get the corresponding digital contents after they have paid the money. They also wish the platform will not reveal their personal privacy. A CDRM scheme can meet the needs of audience, but it may not satisfy the interests of digital content producers. If using the current DDRM scheme, creators of these digital contents will spend a lot of time on encryption, propaganda, sale, authorization, etc., which will inevitably affect their production of high quality digital content. In the long run, this is not good for them.

A. Related Works

Rosenblatt *et al.* [8] in 2002 first proposed their DRM scheme, and gave detailed descriptions of media rights, rights models, principles of DRM systems, DRM standards, and selected commercial solutions. Subsequently, different researchers used different technologies to improve and perfect the existing DRM schemes.

Manuscript received May 6, 2020; revised October 13, 2020 and January 16, 2021; accepted March 1, 2021. Date of publication March 26, 2021; date of current version December 9, 2021. This work was supported in part by the Natural Science Foundation of China under Grant 61303217 and Grant 61502372 and in part by the Guangxi Key Laboratory of Cryptography and Information Security under Grant GCIS201802. *(Corresponding author: Juntao Gao.)*

Juntao Gao is with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China (e-mail: jtgao@mail.xidian.edu.cn).

Haiyong Yu is with the State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China (e-mail: 279980458@qq.com).

Xiuqin Zhu is with the Beijing Qunar Software Technology Company Limited, Beijing 100080, China (e-mail: 895046476@qq.com).

Xuelian Li is with the School of Mathematics and Statistics, Xidian University, Xi'an 710071, China, and also with the Guangxi Key Laboratory of Cryptography and Information Security, Guilin 541004, China (e-mail: xuelian202@163.com).

Digital Object Identifier 10.1109/JSYST.2021.3064356

Yen *et al.* [9] proposed a DRM scheme. The rights owner uploads the data to the agent. The consumer confirms the digital content through the preview permission and completes the transaction through the agent. The agent can track user's permissions through the client.

Ibrahim *et al.* [10] proposed a secure and robust enterprise DRM protocol with efficient storage. The rights owner encrypts the data using the symmetric key, encrypts the symmetric key with the full public key, and signs the ciphertext by using the distributed share and sends it to the authorization server. The authorization server runs the information dispersal algorithm to send the separated ciphertext to the corresponding storage server. And using the verifiable distributed threshold decryption protocol to recover the symmetric key.

Soliman *et al.* [11] proposed an improved enterprise DRM scheme. This scheme separates the data ciphers by using a distributed storage and an information dispersal algorithm. This scheme is similar to that of Ibrahim *et al.*, except that they use the zero-knowledge proof to confirm the validity of the segmented ciphertext stored by the storage server.

Mishra [12] proposed an accountable privacy architecture for DRM system. The rights owner encrypts each file with a different symmetric key, sends these symmetric keys to the license server, sends the ciphertext to the distributor by using a secure channel. The consumer submits a license request to the distributor, and the distributor verifies the permissions and charges the fee. After the license server receives the distributor's message, it sends the license to the consumer.

Zhang *et al.* [13] proposed a novel DRM mechanism on peer-to-peer streaming system. The server and the node authenticate each other by using the key exchange protocol. Confirming the authorized user by updating the owner list, and confirming that the current user is authorized by the watermark in the file.

Wang *et al.* [14] designed a DRM mechanism based on blockchain technology. The rights owner encrypts the data and saves the key, and sets the price and the key usage by using the smart contract method. The buyer obtains the data ciphertext through off-chain, and the buyer selects the required rights and the usage of the key from the smart contract.

Zhang and Zhao *et al.* [15] proposed a novel DRM scheme in P2P networks based on Bitcoin system. The rights owner encrypts the data by using a symmetric key and encrypts the symmetric key by using an RSA public key. The rights requester obtains the ciphertext of the data in a peer-to-peer manner, and obtains the corresponding private key by using the Bitcoin transaction script.

Ma *et al.* [16] proposed a scheme called secure DRM scheme based on blockchain with high credibility. Although the scheme stores data index and authorization information in blockchain, it stores the data plaintext by using a trusted third party.

Zhang and Zehao [17] proposed a DRM mechanism based on blockchain technology. In their scheme, the user needs to register on the blockchain and obtain a certain virtual currency for the transaction.

Through the above-mentioned existed schemes' analysis, there are many comments and difference comparison statement of the related works. We make a small summary of problems in the current DRM schemes as follows.

1) In the current CDRM schemes [9], [12], [16], different users delegate full authority of digital content to the same third party. Although this method greatly facilitates the user, the third party knows the plaintext of the digital content. If the third party has any malicious behavior, it will definitely harm the user's interests. Moreover, if a third party has a system failure, it will inevitably affect the sale of rights and may even disclose useful information of digital content.
2) In the current DDRM schemes [10], [11], [15], the sale of copyright is done directly by the rights owner and the rights requester, without the involvement of a third party. Although this method can better guarantee the confidentiality of digital content, it often significantly increases the user's time overhead and management costs.
3) In the current DDRM schemes based on blockchain [14], [17], the entire DRM system is placed on the blockchain. Although this can ensure the fairness of copyright transactions by using the transparent nature of the blockchain, it will expose too much user's privacy to keep all the interaction records on the blockchain.

B. Our Contribution

In view of the problems in the current DRM schemes, we proposed a dual-blockchain-based DRM scheme with multiauthority ciphertext-policy attribute-based encryption (MA-CPABE) and proxy re-encryption. Our contributions are as follows.

1) A new and efficient MA-CPABE scheme is given to guarantee the confidentiality of digital contents and the flexibility of the digital rights managements. The digital content encrypted by the CPABE is sent to the agent, hence, the agent cannot obtain any information on the digital content because the set of attributes of the agent key does not matches the attributes of the encrypted content. The fine-grained access control policy in CPABE makes the digital rights distribution and management more flexible. Besides, the feature of the multi authority of CPABE greatly reduces the risk of single point of failure. Our scheme can provide indistinguishability of plaintext under adaptively chosen plaintext attack (IND-CPA) security, and can resist collusion attacks. We have an extensive simulation for our scheme, and the result shows that our scheme is very efficient compared to other MA-CPABE schemes.
2) In our system, the agent is not capable of accessing the digital content. His/her main responsibility is to negotiate the access policy with the rights owner, and to re-encrypt the original ciphertext to obtain the new ciphertext with new access policies. The agent is responsible for propagandizing the digital rights to all audience. This greatly reduces the rights owner's time and management overhead.
3) We introduced two blockchains to ensure the DRM trade's fairness. The first one implements the decryption key trading by using the smart contract of Ethereum. The second one is used for the information storage, i.e., we put the abstract of the digital content, the signature and the hash value of ciphertext in the second blockchain. The smart contract is generated by the rights requester with his global public key. The rights owner and the agent can accomplish the smart contract to get the benefits. Then, the rights requester will obtain the partial decryption keys through the smart contract. The smart contract ensures that all the parties in the transaction will get what they need or they will get nothing if any of them violates the protocol.

Since the blockchain only records the rights owner's signature and Ethereum provides all participants' anonymity in smart contracts, our scheme will better protect the user's privacy than the existing schemes. Besides, compared to storing all of digital contents in blockchain, our scheme can reduce the storage cost and enhance the blockchain's efficiency.

The proposed scheme can be applied to many specific scenarios. For example, the scheme could reduce the time cost and management by artists when ensuring fairness of interest between them, agents and fans in the purchase of music copyright transactions, and the members' privacy will be protected meanwhile they enjoy the sharing rights of broadcast of popular TV series.

The rest of the article is organized as follows. We introduce the preliminaries in Section II. Section III presents the overview of our scheme. Section IV is the details of our scheme. Security, performance and efficiency of the scheme are given in Section V. Finally, Section VI concludes this article.

II. Preliminaries

A. Bilinear Pairing

The pairing-based cryptography is based on the discrete logarithm problem. The bilinear pairing has three properties: bilinear, nondegenerate, computable.

Suppose G_1, G_2, and G_T are three multiplicative cyclic groups. The map is defined that $e : G_1 \times G_2 \to G_T$, which satisfies the following three properties.

1) *Bilinear:* For $\forall g_1 \in G_1, \forall g_2 \in G_2, \forall a, b \in Z_p$, equation $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$ holds.
2) *Nondegenerate:* For $\forall g_1 \in G_1, \forall g_2 \in G_2$, there exists $e(g_1, g_2) \neq 1_{G_T}$.
3) *Computable:* There always a valid algorithm to calculate $e(g_1, g_2)$, for $\forall g_1 \in G_1, g_2 \in G_2$.

B. Linear Secret Sharing Scheme (LSSS)

Shamir's secret sharing [18] is an encryption algorithm, in which the secret is divided into multiple parts so that each participant has its own unique part. When it is necessary to recover the secret, you only need to collect enough parts to calculate the shared secret by the decryption algorithm.

LSSS is a general generalization of Shamir's secret sharing scheme. The common description of LSSS is as follow: Let $\mathbf{U}$ be the attribute universe. A secret sharing scheme Π over a set of users P is called linear (over Z_p) if the following conditions hold:

1) the shares for each party from a vector over Z_p^n;
2) there exists a matrix M called the share-generating matrix for Π.

The matrix M has m rows and n columns. For $i = 1, \ldots, m$, the ith row M_i of M is labeled by a party $\rho(i)$, where ρ is a function from $\{1, \ldots, m\}$ to P. Given a column vector $\overrightarrow{v} = (s, v_2, \ldots, v_n)$, where $s \in Z_p$ is the secret to be shared and $v_2, \ldots, v_n \in Z_p$ are randomly chosen. And $\lambda_i = \overrightarrow{v} \cdot M_i$ is the share owned by the ith member.

Any LSSS defined as previously enjoys the linear reconstruction property defined as follows. Suppose that Π is an LSSS for access structure $\mathbf{A}$. Let $S \in \mathbf{A}$ be an authorized set, and $I \subset \{1, \ldots, m\}$ be defined as $I = \{i : \rho(i) \in S\}$. There exist constants $\{\omega_i \in Z_p\}_{i \in I}$ satisfying $\sum_{i \in I} \theta_i M_i = (1, 0, \ldots, 0)$, so that if $\{\lambda_i\}_{i \in I}$ are valid shares of any secret s according to Π, then $\sum_{i \in I} \theta_i \lambda_i = s$. Furthermore, these constants $\{\theta_i\}_{i \in I}$ can be found in time polynomial in the size of the share-generating matrix M. For any unauthorized set, no such constants exists. The LSSS is denoted by (M, ρ).

C. Complexity Assumption

Decisional parallel bilinear Diffie–Hellman exponent assumption (q-parallel BDHE assumption):

Choose a group $\mathbf{G}$ of prime order p according to the security parameter. Let $a, s, b_1, \ldots, b_q \in \mathbf{Z}_p$ be chosen at random and g be a generator of $\mathbf{G}$. If an adversary is given

$$\overrightarrow{y} = g, g^s, g^a, \ldots, g^{(a^q)}, g^{(a^{q+2})}, \ldots, g^{(a^{2q})}$$
$$\forall_{1 \leq j \leq q} g^{s \cdot b_j}, g^{a/b_j}, \ldots, g^{a^q/b_j}, g^{a^{q+2}/b_j}, \ldots, g^{a^{2q}/b_j}$$
$$\forall_{1 \leq j,k \leq q, k \neq j} g^{a \cdot s \cdot b_k / b_j}, \ldots, g^{a^q \cdot s \cdot b_k / b_j}.$$

It must remain hard to distinguish $e(g, g)^{a^{q+1} s} \in \mathbf{G}_T$ from a random element in $\mathbf{G}_T$.

An algorithm $\mathbf{B}$ that outputs $z \in \{0, 1\}$ has advantage ε in solving decisional q-parallel BDHE assumption in $\mathbf{G}$ if

$$|\Pr[\mathbf{B}(\overrightarrow{y}, T = e(g, g)^{a^{q+1} s}) = 0] - \Pr[\mathbf{B}(\overrightarrow{y}, T = R) = 0]| \geq \varepsilon.$$

D. Access Tree Structure

Access Tree Structure: Access tree structure is a common structure, which used to represent access control policy in ABE scheme system.

Let Γ be an Access tree structure. Each nonleaf node of the tree represents a threshold gate, then $0 < k_x \leq \text{num}_x$:

1) when $k_x = 1$, the threshold gate is an OR gate;
2) when $k_x = \text{num}_x$, the threshold gate is an AND gate;
3) when $1 < k_x < \text{num}_x$, it is represents a threshold of $k_x - \text{num}_x$.

The three functions are defined in the access tree structure as follows.

1) *parent(x):* only if x is a leaf node, this function returns the parent of the node x in the access tree.
2) *att(x):* only if x is a leaf node, this function denotes the attribute associated with the leaf node x in the access tree.
3) *index(x):* The access tree also could define an ordering of the subnodes of nonleaf nodes, which these subnodes are numbered from 1 to num. therefore, this function returns the serial number of the node x.

The definition that the attribute set satisfy an access tree structure as follows: Let Γ be an access tree with root node r. Denote by Γ_x the subtree of Γ rooted at the node x. Hence, $\Gamma = \Gamma_x$. $\Gamma_x(\gamma) = 1$ is denoted that a set of attributes γ satisfies the access tree Γ_x. We compute $\Gamma_x(\gamma) = 1$ recursively as follows.

1) If x is a nonleaf node, calculate $\Gamma'_x(\gamma)$ for all children nodes x' of node x. If and only if there are at least k_x children return $\Gamma'_x(\gamma) = 1$, then $\Gamma_x(\gamma)$ returns 1.
2) If x is a leaf node, If and only if $\text{att}(x) \in \gamma$, then $\Gamma_x(\gamma)$ returns 1.

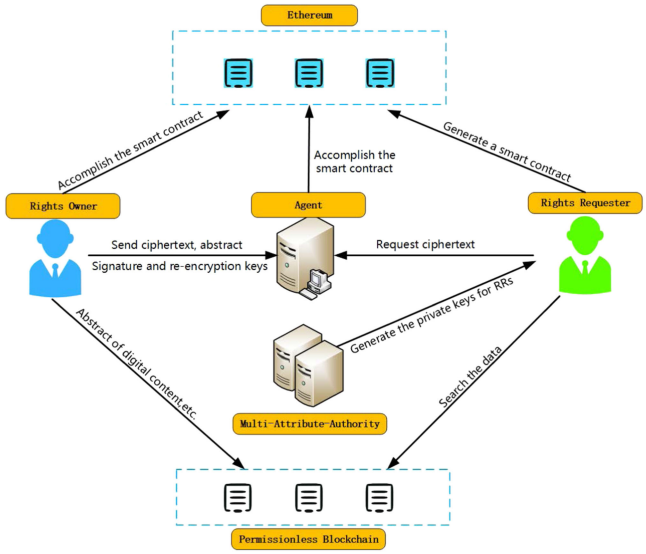

Fig. 1. Overview of our scheme.

III. Overview of Our Proposed Scheme

A. System Model

As described in the Fig. 1, our scheme includes six entities: rights owner (RO), rights requester (RR), the multiattribute-authority (AA), the agent, Ethereum and another blockchain (storing blockchain, a permissionless blockchain). In our scheme, we assume that the RO and the agent are honest, they will fill the correct decryption keys into the smart contract.

1) Rights Owner: In order to save time and computing overhead, the RO sells the rights through an agent. The RO encrypts the data, sends the ciphertext, the signature and the abstract of the digital content to the agent, and posts the abstract, the signature of the ciphertext, the hash value of the symmetric key and the ciphertext on the blockchain. When the RR generates a smart contract on the Ethereum to ask for the rights, the RO generates the partial decryption keys and accomplish the smart contract with the generated decryption keys and his signature.

2) Rights Requester: The RR searches the digital rights by browsing the abstract on the blockchain. Then, the RR generates a smart contract on the Ethereum with his global public key to purchase the partial decryption keys generated by RO and the agent. After he obtains the decryption keys generated by the RO and the agent, he is able to access the digital content by decrypting the ciphertext downloaded from the agent.

3) Multiattribute-Authority: Multiauthority was introduced to our scheme. Each AA is an independent entity. First, they generate their own public and private keys, then manage each user. What is more, each AA also generates a decryption key for the user. AA_K denotes as one of the multiple attribute authorities. On the one hand, this technology could reduce the security risk, which the whole digital right system maybe paralyzed when single-authority is attacked. On the other hand, multiple authorities could reduce the burden of attributes management and key distribution. Besides, it is convenient to meet the demand for distribute system in real life.

4) Agent: The agent is responsible for negotiating the access policy with the RO and using the re-encryption key to re-encrypt the original ciphertext. When the RR generates a smart contract on the Ethereum to ask for the rights, the agent generates the partial decryption keys and accomplish the smart contract with the generated decryption keys and his signature.

5) Ethereum: In order to ensure the fairness, we use the smart contract on the Ethereum to trade the partial decryption keys. We add the signature of the abstract in the smart contract to associate the smart contract with the information on another blockchain. This allows the user's purchase records to be associated with the corresponding digital content, so as to clearly verify that the copyright purchase transaction result is consistent with the digital content and avoid infringement of the rights of buyers.

6) Another Blockchain-Storing Blockchain: The blockchain is used as a ledger for the information of digital content, including the abstract of the data, the hash value of the ciphertext and the signature. The permissionless blockchain is a reasonable choice for determining to store those information of digital content, which is visible to everyone, and therefore, this blockchain is accessible and convenient for all RR from all over the world to search and query what they are interested in. (The blockchain appearing below is this blockchain unless otherwise specified.)

Our scheme consists of eight polynomial time algorithms defined as follows.

1) Setup$(1^\lambda, \mathbf{U}) \rightarrow PP$: Picking a security parameter 1^λ as input, the algorithm outputs the public parameters PP. And the user generates his global key pairs by his private key.

2) AASetup$(PP) \rightarrow (SK_{\mathrm{AA}_K}, PK_{AA_K})$: Taking the public parameter PP as input, the algorithm outputs the secret key and public key pairs $(SK_{\mathrm{AA}_K}, PK_{\mathrm{AA}_K})$ for the authority.

3) KeyGen$(PP, SK_{AA_K}, pk_u, S_u) \rightarrow SK_j$: Taking the public parameter PP, the secret key of the authority AA_K, the user's global public key pk_u and the user's attribute set S_u as inputs, the algorithm outputs the decryption key SK_j corresponding to the attribute set S_u.

4) Encrypt$(PP, PK_{\mathrm{AA}_K}, k, (\mathbf{A}, \rho)) \rightarrow CT$: Taking the public parameters PP, the public key of the authority AA_K, the symmetric key k which is used to encrypt the data m and the access policy $(\mathbf{A}, \rho)$ as input, the algorithm outputs the ciphertext CT with respect to the symmetric key k.

5) Re $-$ KeyGen$(PP, PK_{\mathrm{AA}_K}, SK_j, (\mathbf{A}, \rho')) \rightarrow rk_c$: Taking the public parameter PP, the public key of the authority AA_K, the decryption key SK_j and a new access policy $(\mathbf{A}, \rho')$ as inputs, the algorithm outputs the re-encryption key rk_c corresponding to the new access policy $(\mathbf{A}, \rho')$.

6) Re $-$ encrypt$(CT, rk_c) \rightarrow CT'$: Taking the ciphertext CT and the re-encryption key rk_c as inputs, the algorithm outputs the re-encrypted ciphertext CT' corresponding to the new access policy $(\mathbf{A}, \rho')$.

7) Decrypt$_{\mathrm{semi}}(SK_j, CT') \rightarrow T$: Taking the decryption key SK_j, which is satisfied with the new access policy $(\mathbf{A}, \rho')$ and the re-encrypted ciphertext CT' as inputs, the algorithm outputs the semidecrypted ciphertext T.

8) Decrypt$_{\mathrm{finally}}(CT, T, sk_u) \rightarrow m$: Taking the re-encrypted ciphertext CT', the semidecrypted ciphertext T and the RO's one-time key for one digital work sk_u as inputs, the algorithm outputs the plaintext of the data m.

As described in Fig. 2, our scheme consists of three following phases: data process, policy negotiation, rights purchase.

1) *Data process:* In this phase, we set up our system and attribute authority. The RO encrypts the digital content, signs the signature for the ciphertext of the digital content and generates an abstract belonging to the digital content. Subsequently, the RO sends the ciphertext, the signature and the abstract to the agent. Then, the RO posts the

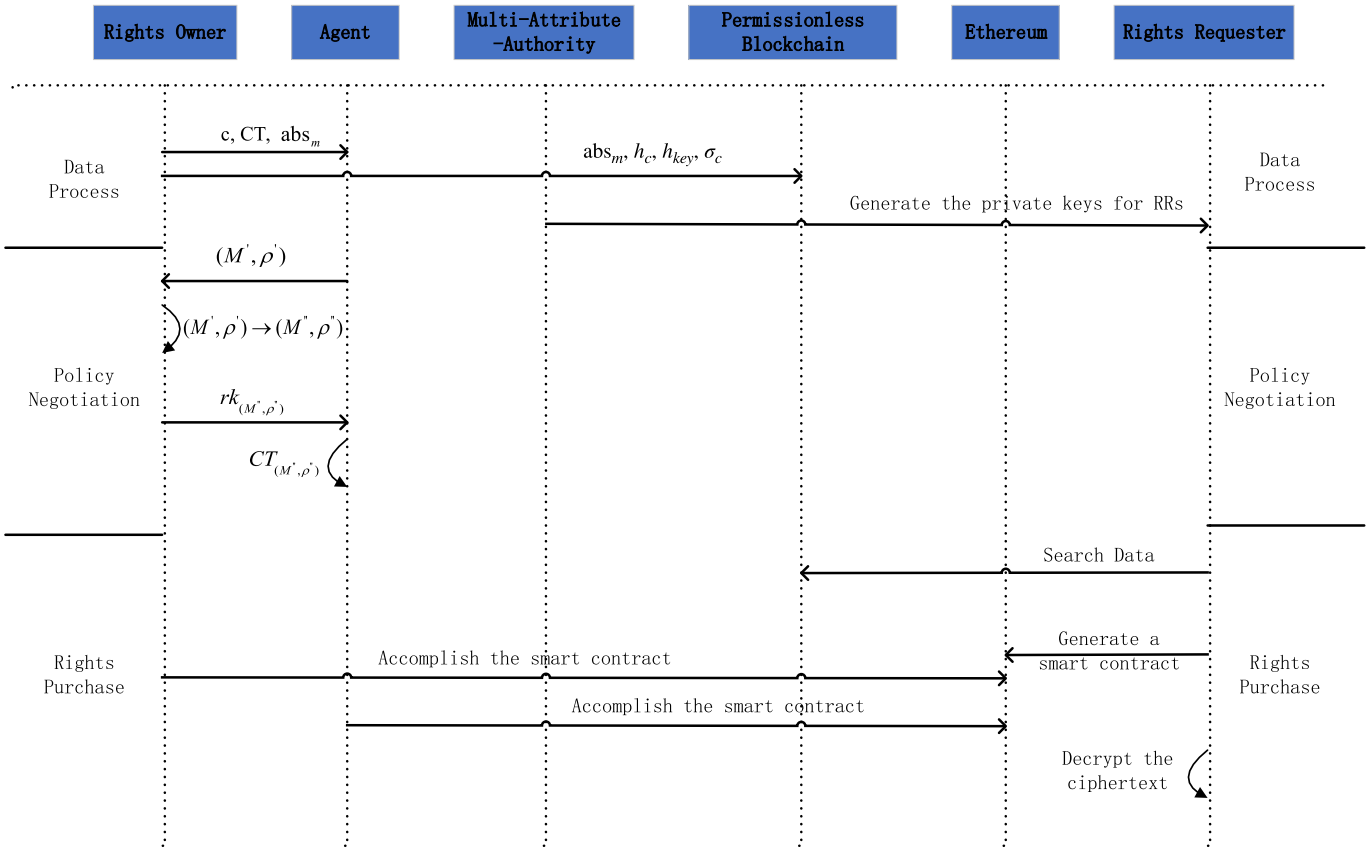

Fig. 2. Framework of the proposed scheme.

abstract, the hash values of the symmetric key and ciphertext on the blockchain.

2) *Policy negotiation:* In this phase, the agent submits an access policy to the RO, and the RO adds some of the attributes that he manages in it to form a new access policy. Then, the RO generates the re-encryption keys, which is corresponding to the new access policy, and sends them to the agent. The agent can use the re-encryption keys to re-encrypt the original ciphertext to generate the re-encrypted ciphertext.
3) *Rights purchase:* In this phase, the RR first searches the digital content, which meets his requirement on the blockchain. Then, the RR generates a smart contract on the Ethereum with his global public key. The RO and the agent generate the partial decryption key with the global public key and accomplish the smart contract with the generated decryption keys and their signatures. Then, the RO and the agent will get the benefits and the RR will obtain the symmetric key through decrypting the re-encrypted ciphertext. Finally, he can decrypt the ciphertext encrypted by the symmetric key to obtain the digital content.

B. Threat Model

In our model, we assume that the agent is honest but curious, and all other entities are not trusted. The agent will honestly execute the sale of the rights and update the access policy, but will be curious about the digital content and find ways to obtain the specific content. Based on the information that the agent will learn, we consider the following threat models.

1) Known Ciphertext Model: This threat model corresponds to the ciphertext-only attack. The agent only knows the ciphertext, the abstract and the signature submitted by the RR, he/she may try to decrypt ciphertext to get more benefits. The RR may also acquire rights by trying to decrypt ciphertext obtained through the agent.

2) Collusion Attack Model: There are two cases in this threat model. The first one is that the RR colludes with each other and try to obtain the rights they do not have. The second one is that the RR colludes with the agent and try to obtain the rights.

C. Security Requirements

Definition 1. IND- CPA.

If for any PPT adversary $\mathcal{A}$, which makes at most probability polynomial time queries in time T, and the advantage $\mathrm{Adv}_{\mathbf{A}}^{\mathrm{IND-CPA}}$ for $\mathcal{A}$ in the following interactive game is at most ε, which is negligible, we say that our scheme has IND-CPA security.

Definition 2. Collusion resistance.

Even if all attribute authorities are corrupted, and the corrupted attribute authority cannot decrypt the ciphertext, we think our scheme can resist collusion attack.

IV. Detail of Our Proposed Scheme

In this section, we propose a new MA-CPABE for the user to transfer digital rights through an agent. And we will describe the three phases of our scheme in detail as follows.

Data process:

1) Setup$(1^\lambda, \mathcal{U}) \rightarrow PP$: Given a security parameter k and the attribute universe $\mathcal{U}$, the setup algorithm runs Setup$(1^\lambda, \mathcal{U}) \rightarrow (g, p, G_1, G_T, e)$, where G_1 and G_T are two multiplicative cyclic groups,g is a generator of group G_1, p is the order of group G_1, and e is a bilinear map, which means $G_1 \times G_1 \rightarrow G_T$. Define two hash functions $H_1 : \{0,1\}^* \rightarrow G_1$ and $H_2 : \{0,1\}^* \rightarrow Z_p^*$. The RO generates $sk_u = H_2(sk_{u-ecc})$, which is a one-time key for one digital work and where sk_{u-ecc} is the private key that the user gets from the blockchain. Then the user generates his global public key $pk_u = g^{1/sk_u}$ by his global private key. The public parameter is $PP = (g, p, e, G_1, G_T, H_1, H_2)$. The user's global key pair is (sk_u, pk_u).

2) AASetup$(PP) \rightarrow (SK_{AA_K}, PK_{AA_K})$: Initialize three kinds of attribute authorities, the normal attribute authority as traditional MA-CPABE scheme, the temporary attribute authority AA_{T_i} by the RO, and the extra attribute authority AA_{E_i} by the agent. The AASetup(PP) algorithm executed by all the attribute authorities chooses two random numbers $\alpha_k, \beta_k \in Z_p^*$ for each AA_K(including the AA_{T_i} and the AA_{E_i}), and chooses a random number $v_x \in Z_p^*$ for the attribute x managed by AA_K. AA_K keeps his private key $SK_{AA_K} = (\alpha_k, \beta_k, \{v_x\}_{x \in S_{AA_K}})$ as a secret, and publicizes his public key $PK_{AA_K} = (g^{\alpha_k}, g^{\beta_k}, \{g^{v_x}\}_{x \in S_{AA_K}})$.

3) KeyGen$(PP, SK_{AA_K}, pk_u, S_u) \rightarrow SK_j$: Every AA_K runs the algorithm KeyGen$(PP, SK_{AA_K}, pk_u, S_u)$, and chooses a random number $r_{j,K} \in Z_p^*$ for each user who has the attribute managed by AA_K and calculates the user's decryption key SK_j

$$SK_j = \begin{cases} D_{j1,K} = g^{r_{j,K}} \\ D_{j1,K,x} = \{(pk_u)^{\frac{\alpha_k}{v_x}} \cdot g^{\frac{\beta_k \cdot r_{j,K}}{v_x}}\}_{x \in S_u} \\ D_{j2,K,x} = \{g^{v_x \cdot r_{j,K}}\}_{x \in S_u}. \end{cases}$$

4) Encrypt$(PP, PP_{AA_K}, k, (\boldsymbol{A}, \rho)) \rightarrow CT$: The RO generates a symmetric key k, and encrypts the data m by the symmetric key to get the ciphertext, i.e., $c = \text{Enc}(k, m)$. That is, c represents the ciphertext of the digital content data m. Then, the RO runs the algorithm Encrypt$(PP, PP_{AA_K}, k, (\boldsymbol{A}, \rho))$ to generates the ciphertext of the symmetric key k. The user chooses a random number $s \in_R Z_p^*$, and generates a vector $\overrightarrow{v} = (s, v_2, v_3, \ldots, v_n)$, where $v_2, v_3, \ldots, v_n \in_R Z_p^*$. The RO chooses an access policy $(\boldsymbol{A}, \rho)$ with which his attributes are satisfied, where $\boldsymbol{A}$ is a matrix of $m \times n$ and ρ is a single mapping function, which maps each row of matrix M to attribute x. The RO calculates the secret share $\lambda_i = \overrightarrow{v} \cdot A_i$, then chooses random numbers $\{r_i \in_R Z_p^*\}_{i \in \{1,2,\ldots,n\}}$ and calculates

$$CT = \begin{cases} C_0 = k \cdot (\prod_{K \in I_A} e(g,g)^{\alpha_k})^s \\ C_1 = g^s \\ C_{i,2} = g^{v_x \cdot \lambda_i} \\ C_3 = g^{\beta_k \cdot s} \\ \sigma_c = \text{sig}(sk_u, (C_0, C_1, C_{i,2}, C_3)) \\ \quad = H_1(C_0, C_1, C_{i,2}, C_3)^{sk_u} \end{cases}$$

where the $K \in I_A$ means that the attribute in the attribute set I_A corresponds to the attribute authority AA_K.

The ciphertext of the symmetric key k is

$$CT = (C_0, C_1, C_{i,2}, C_3, \sigma_c).$$

Then, RO generates an abstract abs_m of the data, and calculates the hash of the symmetric key $h_{\text{key}} = H_1(\text{key})$, the signature of the ciphertext σ_c and the hash of the ciphertext $h_c = H_1(c_{(\boldsymbol{A},\rho)})$. The RO sends c, CT, and abs_m to the agent, and records abs_m, h_{key}, σ_c and h_c on the blockchain. The RO sends the price \$ to the agent.

Policy negotiation:

The RO and the agent negotiate a new access policy $(\boldsymbol{A}', \rho')$. The new access policy contains three parts attributes: the attributes corresponding to the person who can purchase the data rights, the attributes managed by the RO, and the attributes managed by the agent. Then, the RO and the agent negotiate the revenue share of digital rights trading. After determining the new access policy $(\boldsymbol{A}', \rho')$ and the revenue share, the agent generates the re-encryption key corresponding to it as follow.

5) Re $-$ KeyGen$(PP, PK_{AA_K}, SK_j, (\boldsymbol{A}', \rho')) \rightarrow rk_c$: The agent runs the algorithm Re $-$ KeyGen$(PP, PK_{AA_K}, SK_j, (\boldsymbol{A}', \rho'))$, and generates two random numbers $k \in_R Z_p, X \in G_1$. Then he calculates first part of the re-encryption key

$$\begin{cases} rk_{j,1} = D_{j1,K,x}{}^{H_2(X)} \cdot g^k \\ rk_{j,2} = D_{j1,K}{}^{H_2(X)} \\ rk_{j,3} = g^{v_x \cdot k}. \end{cases}$$

Then, he runs the algorithm Encrypt(PP_{AA_K}, PP, k) to generate the ciphertext of the random number X. The user chooses a random number $s' \in_R Z_p^*$, and generates another vector $\overrightarrow{v}' = (s', v_2', v_3', \ldots, v_n')$, where $v_2', v_3', \ldots, v_n' \in_R Z_p^*$. The user chooses another access policy $(\boldsymbol{A}', \rho')$, which is negotiated with the agent. The user calculates the secret share $\lambda_i' = \overrightarrow{v}' \cdot \boldsymbol{A}_i'$, then chooses random numbers $\{r_i' \in_R Z_p^*\}_{i \in \{1,2,\ldots,n\}}$ and calculates

$$\begin{cases} C_0' = X \cdot \left(\prod_{K \in I_A'} e(g,g)^{\alpha_k}\right)^{s'} \\ C_1' = g^{s'} \\ C_{i,2}' = g^{v_x \cdot \lambda_i'} \\ C_{i,4}' = g^{r_i'} \\ C_{i,5}' = g^{\beta_k \cdot \lambda_i'} g^{v_x \cdot r_i'}. \end{cases}$$

The re-encryption key

$$rk = (rk_{j,1}, rk_{j,2}, rk_{j,3}, C_0', C_1', C_{i,2}', C_{i,4}', C_{i,5}').$$

6) Re $-$ encrypt$(CT, rk) \rightarrow CT'$: The agent runs the algorithm Re $-$ encrypt(CT, rk), and gets the part of the ciphertext corresponding to the new access policy by

$$C_r' = \left(\prod_{K \in I_A} e(rk_{j,1}, C_{i,2})^{\theta_i} / (e(rk_{j,2}, C_3) \cdot e(rk_{j,3}, C_1))\right)$$
$$= \left(\left(\prod_{K \in I_A} e(pk_u, g)^{\alpha_k}\right)^s\right)^{H_2(X)}$$

where the θ_i is the value determined by LSSS, and $\sum_{\rho(i) \in x} \theta_i \cdot \lambda_i = s$. Then, the agent gets the ciphertext corresponding to the access policy: $(\boldsymbol{A}', \rho')$, $CT' = (C_0', C_1', C_{i,2}', C_{i,4}', C_{i,5}', C_r')$.

Rights purchase:

The RR searches the digital content on the blockchain. The RR sends a request for the rights to the agent. The agent sends the ciphertext c, CT' and the price \$ $= \$1 + \$2$(\$1 is the benefit for the RO and \$2 is the benefit for the agent.) to the RR. The RR generates a smart contract of Ethereum to purchase the partial decryption keys from the RO and the agent through a smart contract described in Fig. 3.

Algorithm1 Function payment()

1. The RR pays $=$1+$2.
2. The RR sets limitation time T.
3. The RR sets his global public key pk_u.
4. The RR sets the condition of the transaction:

a) The RO submits the one-time key for one digital work sk_u and the partial decryption key generated by the pk_u, the key is corresponding to the attributes managed by the RO;
b) The RO signs the abstract corresponding to the digital content, and submits the signature;
c) The agent submits the partial decryption key generated by the pk_u, the key is corresponding to the attributes managed by the agent;
d) The agent signs the abstract corresponding to the digital content, and submits the signature

Algorithm2 Function transfer()

1. **if** Assert current time $T_1 < T$ **then**
   a) Verify the condition that the RO has submitted the one-time key for one digital work sk_u and the partial decryption key;
   b) Verify the signature generated by the RO;
   c) Verify the condition that the agent has submitted the partial decryption key;
   d) Verify the signature generated by the agent;
   e) Send $1 to the RO and send $2 to the agent.
2. **else**
   a) Send $ to the RR.
3. **end if**

Fig. 3. Details of the smart contract.

The RO, as a temporary attribute authority AA_{T_i}, chooses a random number $r_{j,T_i} \in Z_p^*$ and generates the decryption keys of the attributes managed by the RO as follows:

$$SK_{T_i} = \begin{cases} D_{j1,T_i} = g^{r_{j,T_i}} \\ D_{j1,T_i,x} = \{(PK_{uid})^{\frac{\alpha_{T_i}}{v_x}} \cdot g^{\frac{\beta_{T_i} \cdot r_{j,T_i}}{v_x}}\}_{x \in AA_{T_i}} \\ D_{j2,T_i,x} = \{g^{v_x \cdot r_{j,T_i}}\}_{x \in AA_{T_i}}. \end{cases}$$

The agent, as an extra attribute authority AA_{E_i}, chooses a random number $r_{j,E_i} \in Z_p^*$ and generates the decryption keys of the attributes managed by the agent as follows:

$$SK_{E_i} = \begin{cases} D_{j1,E_i} = g^{r_{j,E_i}} \\ D_{j1,E_i,x} = \{(PK_{uid})^{\frac{\alpha_{E_i}}{v_x}} \cdot g^{\frac{\beta_{E_i} \cdot r_{j,E_i}}{v_x}}\}_{x \in AA_{E_i}} \\ D_{j2,E_i,x} = \{g^{v_x \cdot r_{j,E_i}}\}_{x \in AA_{E_i}}. \end{cases}$$

The RO and the agent use the generated decryption keys and the signature to accomplish the smart contract. The RO and the agent will get the benefits, and the RR will receive the decryption keys generated by them. Then, the RR decrypts the ciphertext with all the decryption keys as follows.

7) Decrypt$_{\text{semi}}(SK_j, CT') \to TT$: The RR sends his decryption key SK_j to the cloud server. If the user's key is satisfied with the access policy $(\boldsymbol{A}', \rho')$, the cloud server can calculate a set of constants $\{\theta_i' \in Z_p^*\}_{i \in \{1,2,\ldots,n\}}$ by LSSS. Then, the cloud server runs the algorithm and calculates the semidecrypted ciphertext (assuming that the cloud server is honest and returns the correct decryption-half results)

$$\begin{aligned} TT &= \left(\prod_K \prod_i \frac{e(C_{i,2}', D_{j1,K,x})}{e(C_{i,5}', D_{j1,K}) / e(C_{i,4}', D_{j2,K,x})} \right)^{\theta_i'} \\ &= \left(\prod_K \prod_i \frac{e(g^{v_x \cdot \lambda_i}', (PK_{uid})^{\frac{\alpha_k}{v_x}} \cdot g^{\frac{\beta_k \cdot r_{j,K}}{v_x}})}{e(g^{r_{j,K}}, g^{\beta_k \cdot \lambda_i}' g^{v_x \cdot r_i}') / e(g^{r_i}', g^{v_x \cdot r_{j,K}})} \right)^{\theta_i'} \\ &= \prod_K e(PK_{uid}, g)^{s' \cdot \alpha_k}. \end{aligned}$$

8) Decrypt$_{\text{finally}}(CT', TT, sk_u) \to m$: The RR can use the semidecryption key TT and one-time key obtain from the RO sk_u to recover the random number by C_0'/TT_u^{sk}. Then, the user can recover the symmetric key k by

$$\begin{aligned} \text{key} &= \frac{C_0}{(C_r'^{(H_2(X))^{-1}})^{sk_u}} \\ &= \frac{k \cdot (\prod_{K \in I_A} e(g,g)^{\alpha_k})^s}{((((\prod_{K \in I_A} e(PK_{uid}, g)^{\alpha_k})^s)^{H_2(X)})^{(H_2(X))^{-1} \ sk_u})} = k. \end{aligned}$$

Finally, the RR can recover the data m by $m = \text{Dec}(k, c)$.

V. Security, Performance, and Effectiveness Analysis

In this section, we will introduce the security, performance and effectiveness analysis of our scheme.

A. Security Analysis

Theorem 1: If the q-parallel BDHE assumption holds, then there is no adversary who can attack us in polynomial time under the chosen plaintext attack model.

Proof: We assume there exists a polynomial-time adversary $\mathcal{A}$ that can inquire the key of polynomial degree in limited time, and attack the IND-CPA game, which we proposed in the security model with an advantage ε. We also assume that the adversary $\mathcal{A}$ chooses two matrices $\boldsymbol{A}^*$ and $\boldsymbol{A}'^*$ of order less than q. And the size of the matrices $\boldsymbol{A}^*$ and $\boldsymbol{A}'^*$ are both $l^* * n^*$ and $l^*, n^* < q$. The adversary $\mathcal{A}$ can ask for any decryption key except that satisfies with the matrices $\boldsymbol{A}^*$ or $\boldsymbol{A}'^*$. Under the above-mentioned restrictions, our security game under multiattribute authority can be equivalent to that under single attribute authority. Next, we construct an adversary $\mathcal{B}$ to solve the q-parallel BDHE assumption with the advantage that cannot be ignored.

Init: The challenger $\mathcal{C}$ uses y, T in the q-parallel BDHE assumption as the challenge input and the adversary $\mathcal{A}$ specifies the set of corrupted authorities $\mathcal{AA}_c$. $\mathcal{A}$ submits two access policies $(\boldsymbol{A}^*, \rho^*)$ or $(\boldsymbol{A}'^*, \rho^*)$, and a global key pair $(sk_u{}^*, pk_u{}^*)$ that he is going to challenge. Assume that $\boldsymbol{A}^*$ and $\boldsymbol{A}'^*$ are matrices of $l^* * n^*$.

Setup: The challenger $\mathcal{C}$ runs the algorithm Setup$(1^\lambda, \mathcal{U}) \to PP$ to generate the public parameters $PP = (g, p, e, G_1, G_T, H_1, H_2)$. For the uncorrupted authorities $k \in (\mathcal{AA} - \mathcal{AA}_c)$, $\mathcal{C}$ chooses a random number a, sets $\alpha_k = \alpha_k' + a^{q+1}$ by letting $e(g,g)^{\alpha_k} = e(g^a, g^{a^q}) \cdot e(g,g)^{\alpha_k'}$ and sets $\beta_k = a + \beta_k'$. For each attribute x, $\mathcal{C}$ randomly chooses a value d_x and sets

$$g^{v_x} = g^{d_x} \prod_{i \in X_p} g^{aM_{i.1}^*/b_i} \cdot g^{aM_{i.2}^*/b_i} \ldots g^{aM_{i.n^*}^*/b_i}$$

where X_p is the indices i set that satisfies $\rho^*(i) = x$. If $X_p = \emptyset$, $g^{v_x} = g^{d_x}$. $\mathcal{C}$ randomly chooses sk_{u-ecc}' for $\mathcal{A}$, sets $sk_u = H_2(sk_{u-ecc}')$ and $pk_u = g^{1/sk_u}$, and sends sk_u and pk_u to $\mathcal{A}$.

Phase 1: In this phase, the challenger $\mathcal{C}$ answers the queries from $\mathcal{A}$ as follow, except that the uid that $\mathcal{A}$ queries is same as the one that he is going to challenge or the set of attributes S_A and the attributes managed by the corrupted authority can satisfy the challenged access structure.

$\mathcal{C}$ chooses a vector $w = (w_1, w_2, \ldots, w_{n^*}) \in Z_p^{n^*}$ and makes $w_1 = -1$. As known by LSSS, there exists such a vector that makes $\boldsymbol{A}_i^* \cdot w = 0$.

$\mathcal{C}$ chooses a random number $r \in Z_p$ and makes $m = r + w_1 a^q + w_2 a^{q-1} + \cdots + w_{n^*} a^{q-n^*+1}$. $\mathcal{C}$ sets $g^{r_{j,K}} = (PK_{uid})^m$, which means $D_{j1,K} = g^{r_{j,K}} = g^{r/sk_u} \prod_{i=1,\ldots,n^*} (g^{a^{q-i+1}})^{w_i/sk_u}$.

As known by the definition above-mentioned that there exists the factor of g^{a^{q+1}/sk_u} in g^{α_k/sk_u}, however, information related to $g^{a^{q+1}}$ is not allowed in the secret key. By observing the definition of $r_{j,K}$, we know that $g^{\beta_k r_{j,K}}$ contains $g^{a^{q+1}}$, which can be eliminated with the factor of g^{α_k}, so we calculate

$$D_{j1,K,x} = (g^{\alpha'_k/sk_u} g^{\beta_k r} g^{\beta'_k w_1 a^q} \prod_{i=2,\ldots,n^*} (g^{a^{q-1}})^{\beta_k w_i/sk_u})^{1/d_x + \frac{a\boldsymbol{A}^*_{i,1} + a^2\boldsymbol{A}^*_{i,2} + \cdots + a^{n^*}\boldsymbol{A}^*_{i,n^*}}{b_i}}.$$

For the $D_{j2,K,x}$, if the attribute x does not satisfy $\rho^*(i) = x$, $D_{j2,K,x} = D_{j1,K} = (PK_{uid})^{m \cdot d_x}$. If the attribute x satisfy $\rho^*(i) = x$, since x is in the access control structure, we must ensure that we cannot simulate the factor $g^{a^{q+1}}$. For $\rho^*(i) = x$, we calculate

$$D_{j2,K,x} = g^{r d_x/sk_u} \prod_{\substack{j=1 \\ i \in X_p}}^{n^*} g^{r a^j \boldsymbol{A}^*_{i,j}/b_i sk_u} \prod_{k=1}^{n^*} g^{d_x a^{q-k+1} w_k/sk_u} \prod_{\substack{j=1 \\ i \in X_p}}^{n^*} \prod_{\substack{k=1 \\ k \neq j}}^{n^*} g^{w_k a^{q-j+k+1} \boldsymbol{A}^*_{i,j}/b_i}$$

$\mathcal{C}$ sends $< D_{j1,K}, D_{j1,K,x}, D_{j2,K,x} >$ to $\mathcal{A}$.

Challenge: $\mathcal{A}$ sends two messages m_0, m_1 of the same length and X_0, X_1 of the same length to $\mathcal{C}$. $\mathcal{C}$ randomly chooses one of the messages m_0, m_1 to generate the original ciphertext: $C_0 = m_b T \cdot \prod_{K \in I_A} e(g^s, g^{\alpha'_k})$, $C_1 = g^s$. Then, $\mathcal{C}$ randomly chooses $y'_1, y'_2, \ldots, y'_{n^*}$, and sets $y'_1 = 0$ to calculate the vector $\overrightarrow{v} = (s + y'_1, sa + y'_2, \ldots, sa^{n^*-1} + y'_{n^*}) \in Z_p^{n^*}$ and the secret share $\lambda_i = \sum_{j=1,\ldots,n^*} (sa^{j-1} + y'_j)\boldsymbol{A}^*_{i,j}$. $\mathcal{C}$ randomly chooses $r'_1, r'_2, \ldots, r'_{n^*}$, we define that R_i means $\rho^*(i) = \rho^*(k)$ for $i \neq k$. Then, $\mathcal{C}$ calculates $C_{i,2} = g^{v_x \lambda_i} = g^{d_x \lambda_i} \prod_{i \in X_p, j \in 1,\ldots,n^*} g^{a^j M^*_{i.j} \lambda_i/b_i}$, $C_3 = g^{\beta_k s} = g^{(a+\beta_k)s}$.

$\mathcal{C}$ calculates $\sigma_c = H_1(C_0, C_1, C_{i,2}, C_3)^{sk_u}$ and sends $< C_0, C_1, C_{i,2}, C_3, \sigma_c >$ to $\mathcal{A}$

$$\begin{aligned} rk_{i,1} &= D_{j1,K,x}{}^{H_2(X)} \cdot g^k \\ &= (g^{\alpha'_k/sk_u} g^{\beta_k r} g^{\beta'_k w_1 a^q} \prod_{i=2,\ldots,n^*} (g^{a^{q-1}})^{\beta_k w_i/sk_u})^{1/\cdot H_2(X)} \cdot g^k \\ rk_{i,2} &= D_{j1,K}{}^{H_2(X)} = g^{r \cdot H_2(X)/sk_u} \cdot \prod_{i=1,\ldots,n^*} (g^{a^{q-i+1}})^{w_i \cdot H_2(X)/sk_u} \\ rk_{i,3} &= g^{v_x k} = g^{d_x k} \end{aligned}$$

$$\cdot \prod_{i \in X_p} g^{a\mathrm{A}^*_{i.1} k/b_i} \cdot g^{a\mathrm{A}^*_{i.2} k/b_i} \ldots g^{a\mathrm{A}^*_{i.n^*} k/b_i}.$$

$\mathcal{C}$ randomly chooses $y''_1, y''_2, \ldots, y''_{n^*}$, and sets $y''_1 = 0$ to calculate the vector $\overrightarrow{v}' = (s' + y''_1, s'a + y''_2, \ldots, s'a^{n^*-1} + y''_{n^*}) \in Z_p^{n^*}$ and the secret share $\lambda'_i = \sum_{j=1,\ldots,n^*} (s'a^{j-1} + y''_j)\boldsymbol{A}'^*_{i,j}$. $\mathcal{C}$ randomly chooses $r''_1, r''_2, \ldots, r''_{n^*}$, we define that R'_i means $\rho'^*(i) = \rho'^*(k)$ for $i \neq k$. Then, $\mathcal{C}$ calculates

$$\begin{aligned} C'_0 &= X_{b'} T \cdot \prod_{K \in I_A} e(g^{s'}, g^{\alpha'_k}), C'_1 = g^{s'} \\ C'_{i,2} &= g^{v_x \lambda'_i} = g^{d_x \lambda'_i} \prod_{i \in X_p, j \in 1,\ldots,n^*} g^{a^j M^*_{i.j} \lambda'_i/b_i} \\ C'_{i,4} &= g^{-r''_i} g^{-s' b_i} \\ C'_{i,5} &= \prod_{j=1}^{n^*} \prod_{i=1}^{n^*} g^{\beta_k (s' a^{j-1} + y'_j)\boldsymbol{A}'^*_{i,j}} \\ &\quad (g^{d_x})^{-r''_i - s' b_i} g^{-r''_i a^j \boldsymbol{A}'^*_{i,j}/b_i} \prod_{k \in R_i} g^{-s' a^j \boldsymbol{A}'^*_{k,j} b_k/b_i}. \end{aligned}$$

Then, $\mathcal{C}$ sends $< rk_{i,1}, rk_{i,2}, rk_{i,3}, C'_0, C'_1, C'_{i,2}, C'_{i,4}, C'_{i,5} >$ to $\mathcal{A}$.

Phase 2. Similar to phase 1.

Guess: $\mathcal{A}$ outputs his guess b^*, if $b^* = b$, $\mathcal{C}$ outputs 0, which means $T = e(g,g)^{a^{q+1}}$, otherwise $\mathcal{C}$ outputs 1, which means T is a random value. If the output T is an element in the group, we have $\Pr[B(y, T = e(g,g)^{a^{q+1}s}) = 0] = 1/2 + Adv_{\mathcal{A}}$, otherwise $\mathcal{A}$ cannot distinguish m_b, and $\Pr[B(y, T = e(g,g)^{a^{q+1}s}) = 0] = 1/2$.

In summary, since there is no effective algorithm in the above-mentioned indistinguishable game, the adversary cannot solve the q-parallel BDHE assumption with a negligible advantage. The challenger wins the game with great advantage. According to our definition in Section III-C, our digital rights management scheme has IND-CPA security.

Correctness: There are two parts to the correctness here, one is the correctness of the re-encrypted ciphertext, and the other is the correctness of the decrypted part. First, we explain the correctness of re-encrypting ciphertext

$$\begin{aligned} C'_r &= \left(\prod_{K \in I_A} e(rk_{j,1}, C_{i,2})^{\theta_i} / (e(rk_{j,2}, C_3) \cdot e(rk_{j,3}, C_1)) \right) \\ &= \left(\prod_{K \in I_A} e(D_{j1,K,x}{}^{H_2(X)} \cdot g^k, g^{v_x \cdot \lambda_i})^{\theta_i} / \right. \\ &\quad \left. (e(D_{j1,K}{}^{H_2(X)}, g^{\beta_k \cdot s}) \cdot e(g^{v_x \cdot k}, g^s)) \right) \\ &= \left(\prod_{K \in I_A} e((PK_{uid}{}^{\frac{\alpha_k}{v_x}} g^{\frac{\beta_k \cdot r_{j,K}}{v_x}})^{H_2(X)} \cdot g^k, g^{v_x \cdot \lambda_i})^{\theta_i} / \right. \\ &\quad \left. (e(g^{r_{j,K} \cdot H_2(X)}, g^{\beta_k \cdot s}) \cdot e(g^{v_x \cdot k}, g^s)) \right) \\ &= \left(\prod_{K \in I_A} e((PK_{uid}{}^{\frac{\alpha_k}{v_x}} g^{\frac{\beta_k \cdot r_{j,K}}{v_x}})^{H_2(X)}, g^{v_x \cdot s}) / \right. \end{aligned}$$

$$e(g^{r_{j,K}\cdot H_2(X)}, g^{\beta_k\cdot s})\Big)$$
$$= \left(\left(\prod_{K\in I_A} e(PK_{uid}, g)^{\alpha_k}\right)^{s}\right)^{H_2(X)}.$$

Then, we explain the correctness of the decryption part. The correctness of the semidecrypted ciphertext

$$TT = \left(\prod_K \prod_i \frac{e(C_{i,2}, {}' D_{j1,K,x})}{e(C_{i,5}, {}' D_{j1,K})/e(C_{i,4}, {}' D_{j2,K,x})}\right)^{\theta_i{}'}$$
$$= \left(\prod_K \prod_i \frac{e(g^{v_x\cdot\lambda_i{}'}, (PK_{uid})^{\frac{\alpha_k}{v_x}}\cdot g^{\frac{\beta_k\cdot r_{j,K}}{v_x}})}{e(g^{r_{j,K}}, g^{\beta_k\cdot\lambda_i{}' g^{v_x\cdot r_i{}'}})/e(g^{r_i{}'}, g^{v_x\cdot r_{j,K}})}\right)^{\theta_i{}'}$$
$$= \left(\prod_K \frac{e(g^{v_x\cdot s'}, (PK_{uid})^{\frac{\alpha_k}{v_x}}\cdot g^{\frac{\beta_k\cdot r_{j,K}}{v_x}})}{e(g^{r_{j,K}}, g^{\beta_k\cdot s'})}\right)$$
$$= \prod_K e(g^{v_x\cdot s'}, (PK_{uid})^{\frac{\alpha_k}{v_x}})$$
$$= \prod_K e(g, PK_{uid})^{s'\cdot\alpha_k}$$
$$= \prod_K e(PK_{uid}, g)^{s'\cdot\alpha_k}.$$

The correctness of the final decryption of ciphertext

$$C_0'/TT_u^{sk} = \frac{X\cdot(\prod_{K\in I_A{}'} e(g,g)^{\alpha_k})^{s'}}{(\prod_K e(PK_{uid}, g)^{s'\cdot\alpha_k})^{sk_u}}$$
$$= \frac{X\cdot(\prod_{K\in I_A{}'} e(g,g)^{\alpha_k})^{s'}}{(\prod_K e(g,g)^{s'\cdot\alpha_k})}$$
$$= X$$

$$key = \frac{C_0}{(C_r'^{(H_2(X))^{-1}})^{sk_u}}$$
$$= \frac{k\cdot(\prod_{K\in I_A} e(g,g)^{\alpha_k})^{s}}{((((\prod_{K\in I_A} e(PK_{uid}, g)^{\alpha_k})^{s})^{H_2(X)})^{(H_2(X))^{-1}})^{sk_u}}$$
$$= \frac{k\cdot(\prod_{K\in I_A} e(g,g)^{\alpha_k})}{(\prod_{K\in I_A} e(PK_{uid}, g)^{\alpha_k})^{sk_u}}$$
$$= \frac{k\cdot(\prod_{K\in I_A} e(g,g)^{\alpha_k})}{(\prod_{K\in I_A} e(g,g)^{\alpha_k})} = k.$$

As can be seen from the formulas in the two above-mentioned parts, our scheme is correct.

Resist collusion attack: Since there are three parties in our scheme: RO: RR; and the agent, we discuss all the possible collusion attacks in our scheme.

The first one is the collusion attack between different RRs. Through the decryption part of our scheme, we can find that RR needs three keys: 1) the key of his own attribute; 2) the key of the attribute managed by the RO; and 3) the key of the attribute managed by the agent. Since different decryption keys are corresponding to different global private keys, even if the RRs combine their decryption keys, these keys cannot be used together.

The second one is the collusion attack between the RR and the agent. The RR needs to get all three parts of the decryption keys to decrypt the ciphertext correctly. Even if the RR collaborates with the agent, the RR simply does not need to purchase the keys of the attribute managed by the agent. We have demonstrated through the security model that the agent cannot obtain any useful information about the digital content through the re-encryption keys and the ciphertext he obtained from the RO. If the RR wants to obtain the corresponding rights, he must purchase the corresponding attribute key from the RO. If the RR and the agent collude with each other, the agent will lose benefits, which has no effect on the RO.

In summary, we say that our scheme has the capability of resisting the collusion attack.

Fairness: Because fine-grained access control is achieved based on CP-ABE and the use of the linear secret sharing scheme, it can be known that if RR wants to decrypt the ciphertext he/she must be obtain all parts of the decryption keys belonging to the RO, the agent and himself (the part of the decryption keys belonging to the RO and agent are obtained through the smart contract transaction). Besides, RR can decrypt ciphertext correctly if only the attribute related to the decryption keys can satisfy the access policy of the ciphertext. Otherwise, the keys that the RR have obtained cannot satisfy the need for decryption.

Through the smart contract, we can see that if the RO and the agent have submitted the partial decryption keys, they will get the benefits and the RR will get the generated decryption keys immediately. In our scheme, the RO and the agent are honest, they will fill the correctly generated decryption keys into the smart contract. With the smart contract of Ethereum and the information of the digital content on the blockchain, it is easy to prove whether the RR has obtained the rights.

In addition, it can be seen that we have added a time lock to the smart contract. When the RO and the agent check the smart contract, they can judge whether the time in the time lock is enough for them to accomplish the smart contract. If not, they can ignore the smart contract. If the time is up and the smart contract is not completed by the RO and the agent, the money deposited by the RR in the smart contract will be returned to the RR.

In summary, we say that our scheme can guarantee the fairness of all parties involved in digital rights transactions.

Privacy protection: We use a blockchain as a ledge for the abstract of the digital content, the signature and the hash values of the symmetric key and the ciphertext. When the user registers on the blockchain, he only needs to provide a pseudonym without providing real identity information. Although we use the smart contract of Ethereum to trade the decryption keys, since the Ethereum is also anonymous, other users of Ethereum can only know some people have made a purchase, but cannot judge who has the corresponding rights in the real world.

But for users who have already purchased the rights, they can prove that they have the corresponding private key by means of zero-knowledge proof, etc., thus proving that they have purchased certain rights.

B. Performance Analysis

In this section, we mainly compare the performance difference between our scheme and the schemes in [9]–[17]. The results of

TABLE I
PERFORMANCE COMPARISON

	Distribute Scheme	Server Load	RO Overhead	Agent knows plaintext	Licensing convenience	Protect the privacy of RP	RO can monitor transactions	RO can deny
[9]	×	high	low	√	hard	×	×	×
[10]	×	high	low	√	easy	×	×	×
[11]	×	high	low	√	easy	×	×	×
[12]	×	high	low	√	hard	×	×	×
[13]	√	low	high	N/A	hard	×	√	√
[14]	√	high	low	N/A	easy	×	√	×
[15]	√	low	high	N/A	easy	√	√	×
[16]	×	high	middle	√	hard	×	√	×
[17]	√	low	low	N/A	easy	×	√	×
Our	√	low	low	×	easy	√	√	×

the comparison are displayed in the Table I. Compared with other schemes, in [9]–[12] are not distributed schemes. The scheme in [13] uses the server to store the authorization list and ciphertexts. Both schemes in [14] and [17] combine smart contracts and blockchains to propose a distributed digital rights management scheme. The scheme in [15] combines a peer-to-peer network with Bitcoin to construct a distributed digital rights management scheme that reduces the load on the server. The scheme in [16] uses a blockchain to store the index of the digital content and the authorization information, but he still handed the digital content to the third party.

We constructed a distributed digital rights management scheme using blockchain. The scheme uses attribute encryption to process data. By encrypting one time, the rights can be sold to multiple consumers, which will reduce the time and computational overhead of RO. Users with attribute private keys can gain access without the user's real identity, which not only improves the convenience of the license, but also protects the privacy of the licensed person. In the scheme, the proxy re-encryption method is used to delegate the authority to the agent for sale, and at the same time, the agent cannot obtain the information of the data itself. A blockchain is used to store copyright purchase records to prevent RO repudiation.

C. Effectiveness Analysis

In this section, we analyze the efficiency of the proposed MA-CPABE scheme. Our MA-CPABE solution is implemented in the Windows system using C++ language, mainly based on the Miracl function library. The download address of the function library is: https://github.com/miracl/MIRACL. The implement platform is a laptop with an Intel Core i3-8100, 3.6 GHz processor and memory 8 GB RAM. The software is configured as: Microsoft Visual Studio 10.0 in a Windows 10 system environment. The MA-CPABE simulation results are shown as follows.

We set the depth k of the access tree to 1, the associated attribute auth required for decryption is set to 10, the attribute managed by each AA is set to 2 and the size of the attribute space $\mathcal{U}$ is increased from 10 to 40. As we can see in the Fig. 4, the most time-consuming step in the algorithm is AASetup, because this step requires generating random numbers associated with each attribute. As the attribute space increases, the time overhead of the step will continue to increase. However, even if the number of attributes in the attribute space is 40, the maximum time of the whole process is only 0.7642 s. If the preparation stage is removed, the time will be shortened to 0.1892 s, which is

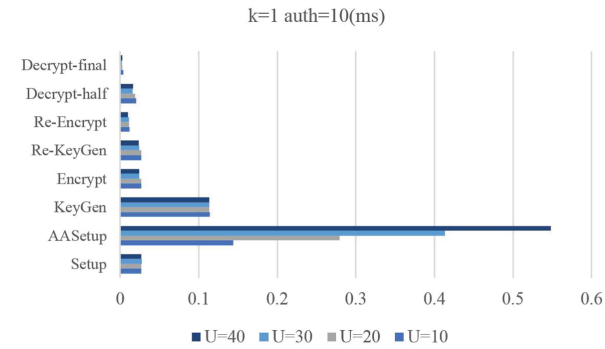

Fig. 4. Efficiency of MA-CPABE in the scheme.

completely within the acceptable range. If you only increase the number of attributes in the attribute space, it will only make the AASetup phase longer, and has little effect on the efficiency of other phases.

Based on the available data, we can speculate that when it is actually used, the system takes a certain amount of time (changing with the size of the attribute space) to initialize. However, in the process of real application, even if the number of attributes related to encryption and decryption is increased, the time for encryption and decryption will only be slightly increased for the user. We can conclude that even if the number of attributes involved in the system is very large (such as a few hundreds), the impact is only the time of system initialization, and will have no effect on the user.

At the same time, we can find from the time overhead of the decrypt-final phase that the real calculations they need to perform for the decryption user only takes very little time because we put most of the bilinear maps in decrypt-half, and this part can be executed by agents such as cloud servers, and without worrying about information leakage. For the consumer, this will make their experience very good.

We fixed the depth of the access tree to 1, changing the number of attributes auth associated with decryption. As can be seen from Fig. 5(a), when the number of decryption-related attributes auth changes from 5 to 10, the time spent in encrypt, re-encrypt, and decrypt-half are only slightly changed, but has little effect on decrypt-final. In the Fig. 5(b), we fixed the number of attributes related to decryption, changed the depth k of the access tree, and the results are basically the same as those obtained from Fig. 5(a).

This means that in the real use process, we can use a more complex access tree (the depth of the tree is deeper) to manage

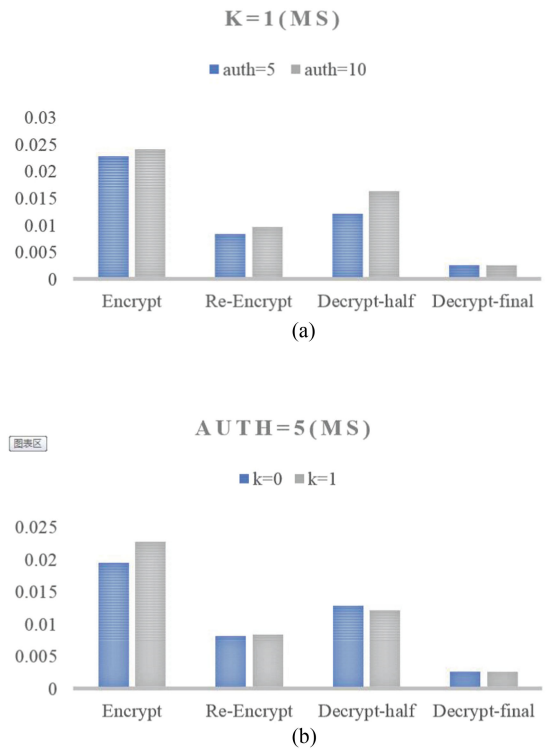

Fig. 5. Efficiency of MA-CPABE in the scheme under different conditions.

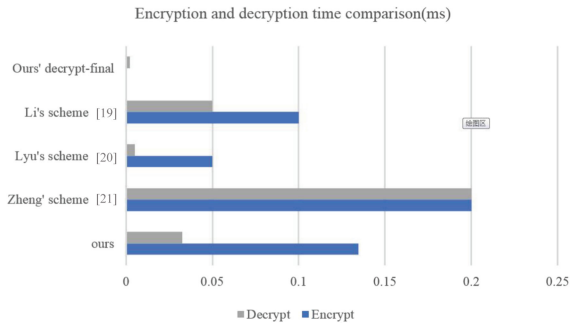

Fig. 6. Efficiency comparison with [19]–[21].

users more finely. At the same time, as the depth of the tree increases, the time and cost of encryption and decryption will only increase slightly, and we do not need to worry too much about the burden of this growth on users.

At the same time, even if the encryption-decryption related attribute is increased (from 5 to 10), the user's encryption and decryption time overhead is only slightly increased. We can speculate that in a real-world scenario, we can use enough relevant attributes to encrypt, which not only improves confidentiality, but also allows finer division of managed users.

These are very meaningful in practical use.

We also compared our scheme with Li's scheme [19], Lyu's scheme [20], and Zheng's scheme [21]. We set the attribute in the attribute space to 20, the number of attributes related to decryption is also 20, and the attribute managed by each AA is set to 10. As can be seen from the comparison results in Fig. 6, the encryption time in our scheme is better than that in Zheng's scheme, but slightly worse than that in Ryu's scheme and Li's scheme. The decryption time in our scheme is better than

Case-1(Data process)		Case-2(Policy negotiate)		Case-3(Rights purchase)	
KeyGen	Encrypt	Re-KeyGen	Re-Encrypt	Semi-Decrypt	Final-Decrypt
(3s+1)E	(l+5)E+P+H	(10+l')E+P+2H	$l' + 2nP$	nE+(3n)P	2E+H

Fig. 7. Calculation costs in our scheme.

that in Zheng's scheme and Li's scheme, but worse than that in Ryu's scheme. However, the decryption in our scheme is divided into two parts. In fact, the part that the user needs to execute is decrypt-final, and the time spent in this stage is better than the above-mentioned two schemes. For the stage decrypt-half, the user can outsource to the cloud service provider. Without the user's global private key, the cloud service cannot obtain any useful information.

We also confirmed the running time of other MA-CPABE schemes in [22]–[26], and compared with them. In [22], when the number of the attributes that managed by each authority is 2, and the number of authority is 10, the encryption time is 5×10^4 ms and the decryption time is 2.5×10^4ms. In [23], when the number of the attributes that managed by each authority is 7, and the number of authority is 4, the encryption time is 135 ms and the decryption time is 83 ms. In [24], when the number of the attributes that managed by each authority is 10, and the number of authority is 2, the encryption time is 150 ms and the decryption time is 110 ms. In [25], when the number of related attributes is 4, the encryption time is 1240 ms and the decryption time is 810 ms. In [26], when the number of the attributes that managed by each authority is 5, and the number of authority is 6, the encryption time is 480 ms, and the decryption time is 1400 ms. All of these schemes have a lower efficiency than ours.

We also consider the calculation cost in our scheme. The related notations used in the calculation cost of the proposed scheme are defined as follows: E denotes an exponential operations in group G, G_T; P denotes a bilinear pairing operations; H denotes a hash function operation; s denotes the number of user's attributes; l denotes the number of attributes in access policy I_A corresponds to the attribute authority AA_K; l' denotes the number of attributes in new access policy I'_A corresponds to the attribute authority AA'_K; n denotes the number of attributes in the decryption key that satisfies the access policy.

Fig. 7 discusses the calculation cost of the proposed scheme, especially including the computation cost of agent (as an intermediate bridge between RR and RO). As can be seen in Fig. 7, the calculation cost of the encryption operation in data process is lower, and the calculation cost of the semidecryption operation in our scheme is higher, but this process is outsourced to a cloud server, i.e., the more complex calculation are done by the cloud and the decryption burden of RR is reduced rapidly.

VI. Conclusion

In order to protect the digital right while reducing the time and computing overhead of the rights owner, we combine the MA-CPABE and proxy re-encryption technology to enable users to sell digital rights through agents without worrying about the agent getting the digital content or revealing information about the digital content. Hence, our scheme effectively ensures the confidentiality of digital content. Furthermore, the smart contract of Ethereum is used to trade the decryption keys generated by the RO and the agent to ensure the fairness of the transaction. The abstract of the digital content, the signature and

hash value of the symmetric key and the ciphertext are posted on another blockchain for the user's convenience to buy and public verification.

We give a detailed analysis of the proposed scheme. Security analysis shows that our scheme can provide IND-CPA security, resists collusion attacks, and protect users' privacy. Besides, our scheme can ensure the fairness through the smart contract of Ethereum. We compared the performance of our scheme with the schemes in [9]–[17], the results show that our scheme can provide more practical features to meet the various needs of users. Moreover, the simulation results show that our scheme has high efficiency under actual conditions.

REFERENCES

[1] N. Kashmar *et al.*, "Smart-AC: A new framework concept for modeling access control policy," *Procedia Comput. Sci.*, vol. 155, pp. 417–424, 2019.

[2] S. Ramamoorthy and B. Baranidharan, "CloudBC—A secure cloud data acess management system," in *Proc. IEEE 3rd Int. Conf. Comput. Commun. Technol.*, 2019, pp. 217–220.

[3] A. Ouaddah *et al.* "Access control in the internet of things: Big challenges and new opportunities," *Comput. Net.*, vol. 112, pp. 237–262, 2017.

[4] A. Ouaddah, A. A. Elkalam, and A. A. Ouahman, "Towards a novel privacy-preserving access control model based on blockchain technology in IoT," in *Proc. Europe MENA Cooperation Advances Inf. Commun. Technologies*, 2017, pp. 523–533.

[5] Y. Zhang, D. Zheng, and R. H. Deng, "Security and privacy in smart health: Efficient policy-hiding attribute-based access control," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 2130–2145, Jun. 2018.

[6] S. Kirrane, A. Mileo, and S. Decker, "Access control and the resource description framework: A survey," *Semantic Web*, vol. 8, no. 2, pp. 311–352, 2017.

[7] D. Servos and S. L. Osborn, "Current research and open problems in attribute-based access control," *ACM Comput. Surveys*, vol. 49, no. 4, pp. 65:1–65:45, 2017.

[8] B. Rosenblatt, B. Trippe, and S. Mooney, "Digital rights management: Business and technology," M&T Press, New York, NY, 2001.

[9] C. T. Yen, H. T. Liaw, and N. W. Lo, "Digital rights management system with user privacy, usage transparency, and superdistribution support," *Int. J. Commun. Syst.*, vol. 27, no. 10, pp. 1714–1730, 2014.

[10] M. H. Ibrahim "Secure and robust enterprise digital rights management protocol with efficient storage," *Int. J. Inf.*, vol. 18,no. 2, pp. 625–640, 2015.

[11] A. H. Soliman, M. H. Ibrahim, and A. E. El-Hennawy, "Improving security and efficiency of enterprise digital rights management," in *Proc. IEEE 6th Int. Conf. Comput., Commun. Netw. Technol.*, 2015, pp. 1–7.

[12] D. Mishra, "An accountable privacy architecture for digital rights management system," in *Proc. 6th Int. Conf. Comput. Commun. Technol.*, 2015, pp. 328–332.

[13] J. Zhang, J. Cai, and Z. Zhang, "A novel digital rights management mechanism on peer-to-peer streaming system," in *Proc. Advances Intell. Inf. Hiding Multimedia Signal Process.*, Springer, 2017, pp. 243–250.

[14] D. Wang *et al.* "A novel digital rights management in P2P networks based on bitcoin system," in *Proc. Int. Conf. Front. Cyber Secur.*, 2018, pp. 227–240.

[15] Z. Zhang and L. A. Zhao, "Design of digital rights management mechanism based on blockchain technology," in *Proc. Int. Conf. Blockchain*, 2018, pp. 32–46.

[16] Z. Ma, W. Huang, and H. Gao, "Secure DRM scheme based on blockchain with high credibility," *Chin. J. Electron.*, 27, no. 5, pp. 1025–1036, 2018.

[17] Z. Zhang and L. A. Zhao, "Design of digital rights management mechanism based on blockchain technology," in *Proc. Int. Conf. Blockchain*, 2018, pp. 32–46.

[18] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[19] X. Li, S. Tang, L. Xu, H. Wang, and J. Chen, "Two-factor data access control with efficient revocation for multi-authority cloud storage systems,"*IEEE Access*, vol. 5, pp. 393–405, 2017.

[20] M. Lyu, X. Li, and H. Li, "Efficient, verifiable and privacy preserving decentralized attribute-based encryption for mobile cloud computing," in *Proc. IEEE 2nd Int. Conf. Data Sci. Cyberspace*, 2017, pp. 195–204.

[21] H. Zheng *et al.*, "Modified ciphertext-policy attribute-based encryption scheme with efficient revocation for phr system," *Math. Problems Eng.*, vol. 2017, pp. 1–10, 2017, Art. no. 6808190.

[22] Q. Li *et al.*, "Secure, efficient and revocable multi-authority access control system in cloud storage," *Comput. Secur.*, 2016, vol. 59, pp. 45–59.

[23] H. S. Gardiyawasam Pussewalageand V. A. Oleshchuk, "A distributed multi-authority attribute based encryption scheme for secure sharing of personal health records," in *Proc. 22nd ACM Symp. Access Control Models Technol.*, 2017, pp. 255–262.

[24] W. Luo and W. Ma, "Efficient and secure access control scheme in the standard model for vehicular cloud computing," *IEEE Access*, vol. 6, pp. 40420–40428, 2018.

[25] C. Pisa, T. Dargahi, A. Caponi, G. Bianchi, and N. Blefari-Melazzi, "On the feasibility of attribute-based encryption for WLAN access control," in *Proc. IEEE 13th Int. Conf. Wireless Mobile Comput., Netw. Commun.*, 2017, pp. 1–8.

[26] Q. Li and H. Zhu, "Multi-authority attribute-based access control scheme in mhealth cloud with unbounded attribute universe and decryption outsourcing,"in *Proc. IEEE 9th Int. Conf. Wireless Commun. Signal Process.*, 2017, pp. 1–7.

[27] B. Waters, "Ciphertext-Policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Proc. Int. Workshop Public Key Cryptography*, 2011, pp. 53–70.

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

[29] M. Chase and S. S. M. Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *Proc. 16th ACM Conf. Comput. Commun. Secur.*, 2009, pp. 121–130.

[30] J. Li *et al.*, "Multi-authority fine-grained access control with accountability and its application in cloud," *J. Netw. Comput. Appl.*, 2018, vol. 112, pp. 89–96.

[31] H. Zhong *et al.*, "Multi-authority attribute-based encryption access control scheme with policy hidden for cloud storage,"*Soft Comput.*, vol. 22, no. 1, pp. 243–251, 2018.

[32] G. Yu *et al.* "Accountable multi-authority ciphertext-policy attribute-based encryption without key escrow and key abuse,"in *Proc. Int. Symp. Cyberspace Saf. Secur.*, 2017, pp. 337–351.

[33] K. Liang, L. Fang, W. Susilo, and D. S. Wong, "A Ciphertext-policy attribute-based proxy re-encryption with chosen-ciphertext security," in *Proc. IEEE 5th Int. Conf. Intell. Netw. Collaborative Syst.*, 2013, pp. 552–559.

[34] Y. Zhang *et al.* "Anonymous attribute-based proxy re-encryption for access control in cloud computing," *Secur. Commun. Netw.*, vol. 9, no. 14, pp. 2397–2411, 2016.

[35] H. Li and L. Pang, "Efficient and adaptively secure attribute-based proxy reencryption scheme," *Int. J. Distrib. Sensor Netw.*, vol. 12, no. 5, 2016, Art. no. 5235714.

[36] X. Zhang and Y. Yin, "Research on digital copyright management system based on blockchain technology," in *Proc. IEEE 3rd Inf. Technology, Networking, Electron. Autom. Control Conf.*, 2019, pp. 2093–2097.

Juntao Gao received the Ph.D. degree in cryptography from the School of Telecommunication and Engineering, Xidian University, Xi'an, China, in 2006.

He is currently an Associate Professor with the School of Telecommunication and Engineering, Xidian University, Xi'an, China. His research interests include pseudorandom sequences and blockchain.

Haiyong Yu received the master's degree from the School of Telecommunication and Engineering, Xidian University, Xi'an, China, in 2020.

His research interest includes the application of blockchain.

Xiuqin Zhu received the master's degree from the School of Telecommunication and Engineering, Xidian University, Xi'an, China, in 2017.

Her research interests include attribute-based encryption.

Xuelian Li received the Ph.D. degree in cryptography from the School of Mathematics and Statistics, Xidian University, Xi'an, China, in 2010.

She is currently an Associate Professor with the School of Mathematics and Staticstics, Xidian University, Xi'an, China. Her research interests include information security and blockchain.