Eng Int Syst (2025) 4: 427–434

Engineering
Intelligent Systems

A Power Data Sharing Scheme Based on Access Policy and Rewritable Blockchain–Access Policies and Power Data Sharing for Rewritable Blockchains

Shu Yang[a]* Ziqiang Zhou[b], Wen Chai[c] and Huanyu Wang[d]

State Grid Shanxi Electric Power Research Institute, Taiyuan, Shanxi, 030001, China

With the rapid development of smart grid technology, data security issues are gradually emerging, especially in terms of confidentiality. Although traditional encryption methods can protect the security of data, they often result in slow processing speed due to low encryption and decryption efficiency, and data is easily tampered with during transmission, posing a serious threat to the security and reliability of power data and the power system. In response to these issues, this article proposes an efficient power data sharing scheme based on Access Control Policy for Rewritable Blockchain Based on Access Control Policy (RBACP). The immutability of blockchain makes power data more secure and transparent. The introduction of access control policies allows only those users who meet the permissions to make changes to data. This article implements an attribute-based encryption This article implements an attribute-based encryption scheme for access control using identity-based signature embedding, resulting in attribute-based encryption for verifiable signatures (ABEVS), which strengthens the verification process, further avoids tampering, and improves data integrity. Finally, this article presents a specific example of RBACP and verifies its security and practicality through experiments and evaluations.

Keywords: smart grid; attribute-based encryption; rewritable blockchain; access strategy

1. INTRODUCTION

With the growing global energy demand and the rapid development of renewable energy technologies, smart grids [1] have become a key technology for energy distribution and trading. Smart grids utilize advanced information technology and network technology to achieve highly intelligent management of energy production, transmission, distribution and consumption. However, the efficient operation of smart grids relies on accurate, real-time data processing, which involves not only energy trading and resource optimization, but also energy data protection and system security enhancement. Blockchain-enabled Internet of Things (IoT)-based edge computing for smart grids has attracted much attention as a promising approach for scaling cloud resources and services [2]. However, the resource-constrained nature of edge nodes makes it difficult to store the entire chain as the volume of sensor-collected IoT data increases.

Attribute-based encryption (ABE)[3] is an advanced encryption technique that allows encryption and decryption operations to be performed on data based on user attributes rather than on traditional public-private key pairs. This technique is a powerful tool used to address data privacy protection and fine-grained access control. Especially in smart grid scenarios, ABE technology allows energy data to be

*Corresponding Author. [a]Email: shuyangtj2014@163.com, [b]zhou5085@126.com, [c]506805818@qq.com, [d]wanghuanyu618@gmail.com

shared and accessed flexibly according to different roles and permissions of users while maintaining a high level of security, greatly improving the flexibility of data management and the overall security of the system.

Blockchain technology[4] has had a profound impact on the field of smart grid research. The core characteristic of blockchain is data immutability, which provides a secure and reliable recording platform for electricity data in smart grids, thus greatly enhancing the trust and transparency of data management. However, this feature of blockchain technology also poses new challenges when it comes to addressing the correction of erroneous data, meeting legal compliance requirements, and protecting data privacy. For example, the EU's General Data Protection Regulation[5] (GDPR), which gives individuals the right to access, correct, and delete their personal data, poses a challenge to traditional blockchain technology.

In order to adapt to the needs of the technology and fulfill the legal requirements, Ateniese et al. [6] proposed the concept of blockchain rewriting for the first time, whereby the traditional hash function is replaced by Chameleon Hash Functions [7]. Chameleon Hashes, CH) generate collisions through the CH technique to achieve rewrite operations, but this method is only applicable to the rewriting of data blocks at coarse-grained level. In order to achieve fine-grained access control rewriting, Derler et al. [8] proposed the concept of Policy-based Chameleon Hash (PCH). essentially, PCHcombines CHET [9] and CP-ABE[10]. The temporary trap key that generates the hash collision is contained in the ciphertext of CP-ABE, and the trap key can be obtained only when the attribute satisfies the access policy, thus enabling transaction-level blockchain rewriting. Much of the subsequent work based on PCH is centered around its usability, such as the mechanism of recourse to users [11], revocability [12,13], decentralization [14,15], one modification operation[16] and k-modification operations[17] etc., the challenge of power data management in smart grids can be solved through the use of cryptography and blockchain's controllable and editable technology. This technology allows users with appropriate permissions to modify power data when necessary, while maintaining the transparency and traceability of modification operations through version control and audit trail mechanisms. In this paper, a controllable and rewritable blockchain scheme is designed from the data management needs of smart grid.

Combining the immutability of blockchain technology with the flexibility and security of attribute-based cryptography, this paper proposes a new type of blockchain technology scheme (Rewritable Blockchain Based on Access Control Policy, RBACP) that can maintain the core advantages of blockchain and provide the flexibility of data modification, which is of great significance for the development of smart grid.

2. PREPARATORY KNOWLEDGE

2.1 Asymmetric Bilinear Mapping[18]

Definition 1: Let $GroupGen$ be a probabilistic polynomial algorithm that takes as input the security parameter 1^{λ} and returns a group $G = (p, G_1, G_2, G_T, e, g_1, g_2)$.

where p is a prime of bits of $\Theta(\lambda)$, $G_1 G_2$ and G_T are cyclic groups of order p, g_1 and g_2 are generators of G_1 and G_2 respectively, and $e{:}G_1 \times G_2 \rightarrow G_T$ is a non-degenerate bilinear map (also known as a pairing) and satisfies the following three properties:

(1) Bilinear: for any $g_1 \in G_1, g_2 \in G_2$, there are; $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$

(2) Non-degeneracy: $\exists g_1 \in G_1, g_2 \in G_2$ makes $e(g_1, g_2) \neq 1$

(3) Computability: are efficient algorithms that can compute $e(g_1, g_2)$ for $\forall g_1 \in G_1, g_2 \in G_2$.

In the special case of$G_1 = G_2$, the bilinear pairing is said to be symmetric; otherwise it is asymmetric.

2.2 n-q-type assumptions[10]

Definition 2: Let $G = (p, G_1, G_2, G_T, e, g_1, g_2)$ be a pairwise group. Randomly choose $(x||\delta||y) \leftarrow Z_p \times Z_p \times Z_p^n$.

Compute,

$D^1 = (g_1, g_1^x, \{g_1^{\delta y[i]}, g_1^{\frac{1}{xy[i]}}\}_{i\in[n]}, \{g_1^{\frac{\delta y[i]}{xy[i]}}\}_{i,j\in[n],i\neq j}) D^2 = (g_2^{\frac{1}{\delta}}, g_2^{\frac{x}{\delta}}, \{g_2^{y[i]}\}_{i\in[n]}) T_0 = e(g_1, g_2)$, and randomly select $T_1 \in G_T$.

The n-q-type assumption means that the advantage $Adv_{D,n}^{q\text{-}type}(1^{\lambda}) = |\Pr[D(p, G_1, G_2, G_T, e, D^1, D^2, T_0) \Rightarrow 1] - \Pr[D(p, G_1, G_2, G_T, e, D^1, D^2, T_1) \Rightarrow 1]|$ is negligible. That is, there is no probabilistic polynomial time attacker, and D able to distinguish between T_0 and T_1 with a non-negligible advantage.

2.3 Chameleon Hashes With Ephemeral Trapdoors CHET[9]

Definition 3: CHET is a variant of the chameleon hash *CH* [7] used to design rewritable blockchains based on access policies. In *CHET*, the security of the hash includes indistinguishability, and collision resistance. Collisions can only be successfully found if the user holds both permanent and temporary trapdoors. A *CHET* scheme with message space $\mathcal{M}$ consists of the following five algorithms:

- $CHET.ParGen(1^{\lambda}) \rightarrow (pp)$: Input security parameter λ, output public parameter pp.
- $CHET.KeyGen(pp) \rightarrow (sk_{CHET}, pk_{CHET})$: Input public parameter pp and output private key sk_{CHET} and public key pk_{CHET}.
- $CHET.Hash(pk_{CHET}, m) \rightarrow (h_{CHET}, r_{CHET}, etd)$: Input public key pk_{CHET} and message $m \in \mathcal{M}$, output hash h_{CHET}, random number r_{CHET} and temporary trapdoor etd.
- $CHET.Verify(pk_{CHET}, m, h_{CHET}, r_{CHET}) \rightarrow (0/1)$: Input the public key pk_{CHET}, the message $m \in \mathcal{M}$, the hash h_{CHET} and the random number r_{CHET}, then determine if (m, h_{CHET}, r_{CHET}) is valid and output a bit.

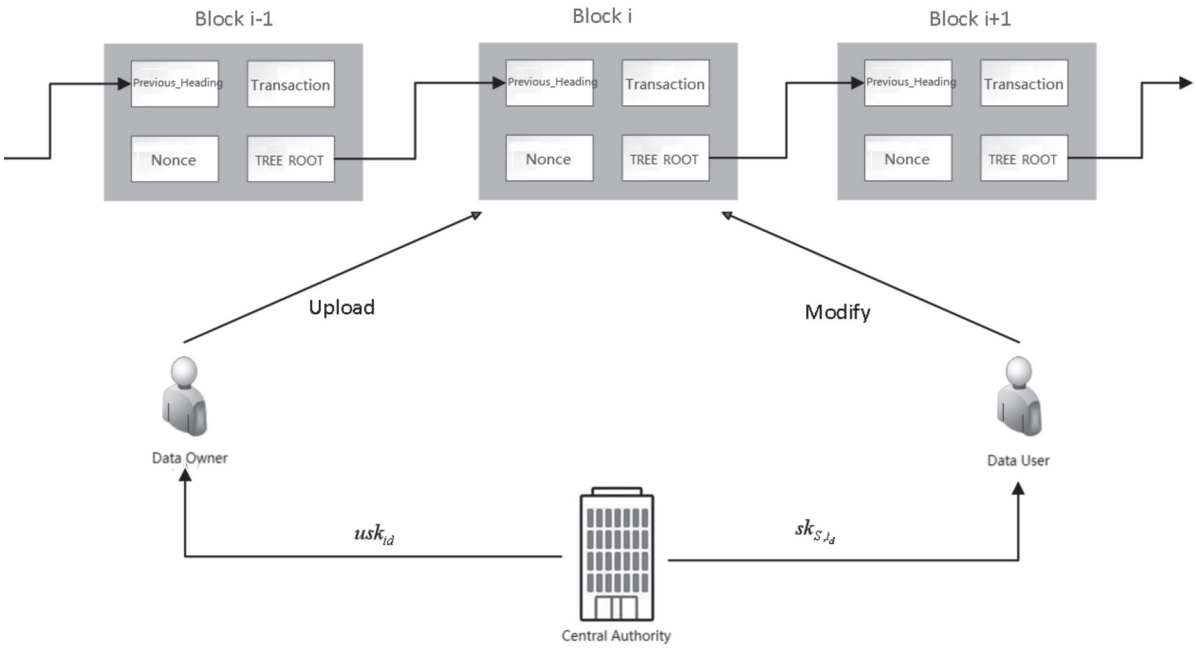

Figure 1 System Model Diagram.

- $CHET.Adapt(sk_{CHET}, etd, m', m, h_{CHET}, r_{CHET}) \rightarrow (r'_{CHET})$: Input a key sk_{CHET}, a temporary trapdoor etd, a message $m' \in \mathcal{M}$, a hash $h_{CHET}$and a random number r_{CHET}, output a random number r'_{CHET}.

2.4 Attribute-Based Encryption for Verifiable Signatures (ABEVS)[10]

Definition 4: ABEVS implements the tracking accountability and revocation of malicious users. The scheme has a message space $\mathcal{M}$, an attribute space U and a user space N and consists of the following four main algorithms:

- $ABEVS.Setup(1^{\lambda}, T) \rightarrow (mpk_{ABEVS}, msk_{ABEVS})$: Input the security parameter λ and the binary tree T, output the public parameter mpk_{ABEVS} and the master key msk_{ABEVS}.

- $ABEVS.KeyGen(msk_{ABEVS}, S, i_d) \rightarrow (sk_S)$: Input the master key msk_{ABEVS}, the user attribute base $S \in U$ and the encryptor identity $i_d \in N$, and output the user's corresponding private key sk_S.

- $ABEVS.Enc(mpk_{ABEVS}, m, (M, \pi), i_d) \rightarrow (ct, \sigma)$:

  Input master public key mpk_{ABEVS}, message $m \in \mathcal{M}$, access policy (M, π) and decryptor identity $i_d \in N$, output ciphertext ct and signature σ.

- $ABEVS.Dec(sk_S, ct, \sigma) \rightarrow (m)$: Enter the key sk_S, the ciphertext ct and the signature σ, and output the plaintext $m \in \mathcal{M}$.

3. PROGRAMMATIC DEFINITIONS

3.1 System Model

The system model diagram for this program is shown in Figure 1. The following are the four entities that make up the system model diagram:

(1) Authorization Center (Central Authority, CA): CA is a fully trusted authority responsible for generating the master key and master public key, while generating private keys for users.

(2) Data Owner (DO): sends a private key request to the CA, sends the hash value of the power data to the blockchain, and determines the range of users who can make changes to the power data.

(3) Blockchain (BC): the BC used to store the hash value of power data.

(4) Data User (DU): sends a private key request to CA, and can modify the data when it meets the access rights to the power data.

3.2 Definition of the Formalization of The Program

- $SetUp(1^{\lambda}) \rightarrow (msk, pk)$: The authority executes the algorithm by entering the security parameter λ and outputting the master private key msk and the public key pk.

- $DOKeyGen(pk, msk, i_d) \rightarrow (usk_{i_d})$: The authority executes the algorithm by inputting the public key pk, the master private key msk and the user identity i_d, and outputting the user private key usk_{i_d}.

- $DUKeyGen(pk, msk, S, i'_d) \rightarrow (sk_{S,i'_d})$:

  The authority executes the algorithm by inputting the public key pk, the master private key msk, the modifier identity i'_d and the set of user attributes S, and outputting the modifier's private key sk_{S,i'_d}.

- $Hash(pk, usk_{i_d}, m, (M, \pi)) \rightarrow (h, r, \sigma_{i_d})$: The data owner executes the algorithm by inputting the master public key pk, the data owner key usk_{i_d}, the data m and the access policy (M, π), and outputting the hash h, the random number r and the signature σ_{i_d}.

- $Verify(pk, m, h, r, \sigma_{i_d}) \rightarrow (0/1)$: Any participant can execute the algorithm by entering the master public key pk, the data m, the hash value h, the random number r and the signature σ_{i_d} corresponding to the identityi_d, outputting 1 if it passes the verification, and 0 otherwise.

- $Adapt(pk, sk_{S,i'_d}, m, m', h, r, \sigma_{i_d}, i'_d) \rightarrow (r', \sigma_{i'_d})$: The data modifier executes the algorithm by inputting the modifier's private key sk_{S,i'_d}, the data m, the modified data m', the hash h, the random number r, the signature of the data owner σ_{i_d} and the modifier's identity i'_d. Output the random number corresponding to the modified data content r' and the modifier's signature $\sigma_{i'_d}$.

4. PROGRAM SPECIFIC CONSTRUCTION

- $SetUp(1^\lambda) \rightarrow (msk, pk)$: The CA enters the security parameters and then performs the following steps:

  (1) Run the group generation algorithm to obtain $\mathcal{G} = (p, G_1, G_2, G_T, e, g_1, g_2)$.

  (2) Randomly select: $x \in Z_p^*$, $h_1 \in G_2$ to get the chameleon key pair $(sk_{CHET}, pk_{CHET}) = (x, h_1^x)$.

  (3) Set the hash functions H_1:$[|U| + 1] \rightarrow G_1$

  H_2:$\{0, 1\}^* \rightarrow Z_p$ and H_3:$\{0, 1\}^* \rightarrow G_1$.

  Get the key pairs $mpk_{ABE} = \{g_1, g_2, e(g_1, g_2)^\alpha, h_1, H_1, H_2, H_3\}$ and $msk_{ABE} = \{\alpha, x\}$ for attribute-based encryption.Output $msk = \{sk_{CHET}, msk_{ABE}\}$ and $pk = \{pk_{CHET}, mpk_{ABE}\}$.

- $DOKeyGen(pk, msk, i_d) \rightarrow (usk_{i_d})$: CA inputs, pk msk and the identity of the DO i_d, then computes $usk_{i_d} = H_3(i_d)^x$, and finally uses usk_{i_d} as the private key of the data owner and outputs it.

- $DUKeyGen(pk, msk, S, i'_d) \rightarrow (sk_{S,i'_d})$: CA enter, pk msk, the set of attributes corresponding to the DU S and the identity of the DM i'_d, and then perform the following steps:

  Randomly select $r \in Z_p$ and compute, $K_1 = g_1^\alpha \cdot H_1(|U| + 1)^r\{K_{2,u} = H_1(u)^r\}_{u \in S}$ and $K_3 = g_2^r$. Calculated by entering the user's identity $K_{i'_d} = msk_{i_d} = H_3(i'_d)^x$.

  (3) Output $sk_{S,i'_d} = (S, K_1, \{K_{2,u}\}_{u \in S}, K_3, K_{i'_d}, i'_d)$.

- $Hash(pk, usk_{i_d}, m, (M, \pi)) \rightarrow (h, r, \sigma_{i_d})$: DO enter pk, the data owner's key usk_{i_d}, the data m and the access structure (M, π), and then perform the following steps:

  Choose the random number $r^* \in Z_p^*$,

  calculate $p = pk_{CHET}^{r^*}$.

  Choose random numbers as ephemeral trapdoors $etd \in Z_p$, compute $e = H_2(etd)$ and $h'_1 = h_1^e$.

  (3) Calculate the hash $h_{CHET} = ph'^m_1$.

  (4) Select random vector $v \in Z_p^{n_2 - 1}$, random numbers, $s_1 s' \in Z_p$, compute, $ct_1 = g_2^{s'}$, $ct_{2,i} = H_1(|U| + 1)^{M_i(s_1||v)^T} \cdot H_1(\pi(i))^{s'} ct_3 = g_2^{s_1} cp = e(g_1, g_2)^{\alpha s_1}$, and $c = cp \cdot etd$ to get ciphertext $ct = (ct_1, \{ct_{2,i}\}_{i \in n_1}, ct_3, c)$.

  (5) Select the random numbers $g' \in G_1$ and $k \in Z_p$, compute $R_1 = e(g', h_1)^k$ and $w = H_2(L)$, where the signature object $L = (m, ct, r^*, p, h'_1, h_{CHET}, R_1)$ and then get the signature $\sigma_{i_d} = (u, w)$.

  (6) Output the hash group

  $h = (ct, p, h'_1, h_{CHET}, H_2, i_d)$, the random number $r = r^*$ and the signature $\sigma_{i_d} = (u, w)$.

- $Verify(pk, m, h, r, \sigma_{i_d}) \rightarrow (0/1)$: Any system user can execute the algorithm to verify that the hash is valid by performing the following steps:

  (1) Verify $h_{CHET} \stackrel{?}{=} pk_{CHET}^{r^*} \cdot h'^m_1$.

  (2) $R'_1 = e(u, h_1) \cdot e(H_3(i_d), (h_1^x)^{-1})^w$.

  (3) Verify $w \stackrel{?}{=} H_2(L)$, where the signature object $L = (m, ct, r^*, p, h'_1, h_{CHET}, R'_1)$.

  (4) Output 1 if all the above validations pass, otherwise output 0.

- $Adapt(pk, sk_{S,i'_d}, m, m', h, r, \sigma_{i_d}, i'_d) \rightarrow (r', \sigma_{i'_d})$: DU inputs pk, the private key of the data modifier sk_{S,i'_d}, the old data m, the new data m' and the outputs of DO (h, r, σ_{i_d}) and then performs the following steps:

  (1) If $Verify(pk_{PCHAR}, m, h, r, \sigma_{i_d}) = 1$ and the user attribute satisfies the access structure, perform the following steps.

  (2) $e(K_1, ct_3) * \frac{e(\prod_{i \in I}(K_{2,\pi(i)})^{\gamma_i}, ct_1)}{e(\prod_{i \in I}(ct_{2,i})^{\gamma_i}, K_3)} = cp$, the existence of $\{\gamma_i\}_{i \in I}$ makes $\sum_{i \in I} \gamma_i M_i = (1, 0, \ldots, 0)$, then finally calculate $etd = c/cp$.

  (3) Calculate $e = H_2(etd)$ and then get $r' = r^* + (m - m') \cdot e/x$.

  (4) Choose random numbers $g' \in G_1$ and $k \in Z_p$, compute $R_1 = e(g', h_1)^k w = H_2(L')$ and $u = (K_{i'_d})^w \cdot (g')^k$, where the signature object $L' = (m', ct, r', p, h'_1, h_{CHET}, R_1)$.

  (5) Output the random number r' and the signature $\sigma_{i'_d} = (u, w)$.

5. SECURITY CERTIFICATES

Theorem: If the n-q-type assumption holds in the $RBACP$ scheme, then the security of $FABEO$ in $RBACP$ is guaranteed so that it can resist the chosen plaintext attack - CPA.

PROOF: Suppose there is an attacker$A$ attacking

CPA secure $RBACP$ with a non-negligible advantage, then there exists a challenger D solving the problem of the n-q-type hypothesis by taking advantage of A. The simple procedure is that D inputs $(p, G_1, G_2, G_T, e, D^1, D^2, T_b)$ and then performs the relevant action of A.

Init: A Select a challenge strategy $\Lambda^* = (M^*, \pi^*)$ and an undo list R^*.

SetUp:D Generates public parameters by performing the following steps:

(1) Implicitly set $\alpha = \frac{\delta}{x}$ and get the public parameter $(p, G_1, G_2, G_T, e, g_1, g_2^{\frac{1}{\delta}}, e(g_1, g_2^{\frac{1}{\delta}})^{\alpha})$, where the last item can be calculated by $e(g_1^{\frac{1}{xy[1]}}, g_2^{y[1]})$.

Phase1: D When simulating a stochastic prediction model, a list L is used to hold all queries and outputs. The list contains elements of the form $(u, H_1(u))$. At the time of querying, first check whether the attribute u is included in the list. If it is in the list, then the returned query results will be consistent with the previous one; otherwise, the output is simulated in the following way. First a new index $\varepsilon_u \in Z_p$ is randomly selected for randomization and then the following three cases are distinguished:

$$H_1(u) = \begin{cases} g_1^{-\frac{1}{xy[1]}} & if\, u = |U| + 1 \\ g_1^{\sum_{j\in[n_2]} \frac{M_{i,j}*}{xy[j]}} \cdot g_1^{\varepsilon_u} & if\, \exists\, i.s.t.\, \pi(i) = u \\ g_1^{\varepsilon_u} & otherwise \end{cases}$$

Finally, add $(u, H_1(u))$ to the list. (g_1 and $g_1^{\frac{1}{xy[1]}}$ can be used, provided by assumption, to count all items). A Submit the user's associated identity and attributes (i_d, S) to D are used to request the associated key. First, compute $w \in Z_p^{n_2}$, for all $i \in [n_1]$, such that $< w, M_i^{T*} >= 0$, $\pi(i) \in S$. The vector is valid and computable, assuming $w[1] = 1$. Randomly select $\gamma \in Z_p$ and then implicitly set $r = \delta \sum_{j\in[n_2]} w[j]y[j] + \gamma$. Finally, the portion of the key used by the user to decrypt the ephemeral trapdoor can be simulated and computed as $K_1 = g_1^{\frac{\delta}{x}} \cdot g_1^{-\frac{\delta \sum_{j\in[n_2]} w[j]y[j]+\gamma}{xy[1]}} = g_1^{-\frac{\sum_{j\in[2,n_2]} \delta w[j]y[j]}{xy[1]}} \cdot g_1^{-\frac{\gamma}{xy[1]}}$ $K_{2,u} = H_1(u)^{\delta \sum_{j\in[n_2]} w[j]y[j]+\gamma}$ and $K_3 = (g_2^{\frac{1}{\delta}})^{\delta \sum_{j\in[n_2]} w[j]y[j]+\gamma} = g_2^{\sum_{j\in[n_2]} w[j]y[j]} \cdot g_2^{\frac{\gamma}{\delta}}$. For $K_{2,u}$, the calculation is based on the three cases of $H_1(u)$.

Challenge: The attacker, A, submits two messages, m_0, m_1, to the challenger D. D performs the following steps based on the originally submitted challenge strategy $\Lambda^* = (M^*, \pi^*)$.

(1) Set $s_1 = x + \tau_1 s' = x + \tau'$ and $v[j] = \frac{xy[1]}{y[j]} + \tau_j$, where $j \in [2, n_2]$. Then, calculate $ct_1 = g_2^{\frac{x}{\delta}} \cdot g_2^{\frac{\tau'}{\delta}}$ $ct_{2,i} = g_1^{-\frac{M_i^*(s_1||v)^T}{xy[1]}} \cdot (g_1^{\sum_{j\in[n_2]} \frac{M_{i,j}*}{xy[j]}} \cdot g^{\varepsilon_{\pi(i)}})^{s'}$ $ct_3 = g_2^{\frac{x}{\delta}} \cdot g_2^{\frac{\tau_1}{\delta}}$ and $cp = T_b \cdot e(g_1^{\frac{1}{xy[1]}}, g_2^{y[1]})^{\tau_1}$.

(2) D flips a coin $u \in \{0, 1\}$, then calculates $c = cp \cdot m_u$.

(3) Finally, D sends the ciphertext used to decrypt the short trapdoor $ct = (ct_1, ct_{2,i}, ct_3, c)$ to the attacker A.

Phase 2: Same as Phase 1.
Guess: A outputs two guesses about uu':

1. If $u = u'D$ outputs a guess for $bb' = 0$. When $b = 0$, $cp = e(g_1, g_2) \cdot e(g_1, g_2)^{\frac{\tau_1}{x}} = e(g_1, g_2^{\frac{1}{\delta}})^{\frac{\delta}{x}(x+\tau_1)} = e(g_1, g_2^{\frac{1}{\delta}})^{\alpha s_1}$, then the attacker A can obtain a legitimate ciphertext.

2. If $u \neq u'D$ outputs a guess for $b\,b' = 1$. When $b = 1$, then cp is uniformly distributed in G_T. Then A loses the attacking advantage.

CONCLUSION: Based on the above, it can be found that if the adversary A breaks the $RBACP$ scheme of CPA security by a non-negligible margin, then a challenger, D, can solve the n-q-type hypothesis problem in polynomial time. However, the n-q-type assumption problem is difficult to solve during computation, so the $RBACP$ scheme in this paper is CPA secure.

6. EXPERIMENTAL EVALUATION

This section uses the Charm framework in Python 3.8 [19]. The scheme RBACP is implemented. the computer configuration for the experiment is Intel(R) Core (TM) i5-12500H 2.50 GHz and 16GB RAM. then a 64-bit ubuntu environment is built in a virtual machine to perform the experiment. We use the MNT224[20] curve for bilinear pairing, which provides a 100-bit security level. We uniformly set the number of attributes and policy size as $\{10, 20, \ldots, 100\}$, and then execute each algorithm of this scheme RBACP 10 times according to the attribute set size. The experimental results are shown in Fig. 2.

In the experiment, the input parameter of SetUp in the initialization phase is a constant so that the running time does not increase with the increase of the number of attributes. The running of both Hash and DUKeyGen is related to the set of attributes, then the execution time of the algorithm increases with the increase of the number of attributes. The Verify algorithm needs only to validate the uploaded hash to the blockchain and the user's signature, which is not related to the attributes [21]. Therefore, as the number of attributes increases, the algorithm running time tends to level off. In the Adapt phase, the decryption operation of ABE needs to be performed first to obtain the conditions for finding collisions. Both the decryption operation and finding collisions have nothing to do with the number of attributes, so the algorithm's running time also tends to be stable.

The operational efficiency of the RBACP scheme is affected mainly by attribute-based encryption. The attribute-based

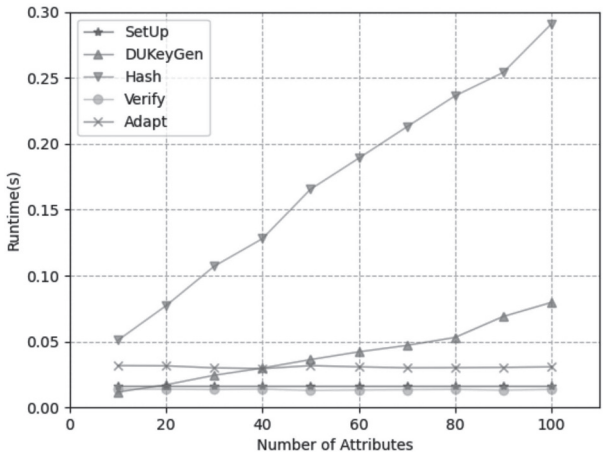

Figure 2 The running time of each algorithm in RBACP.

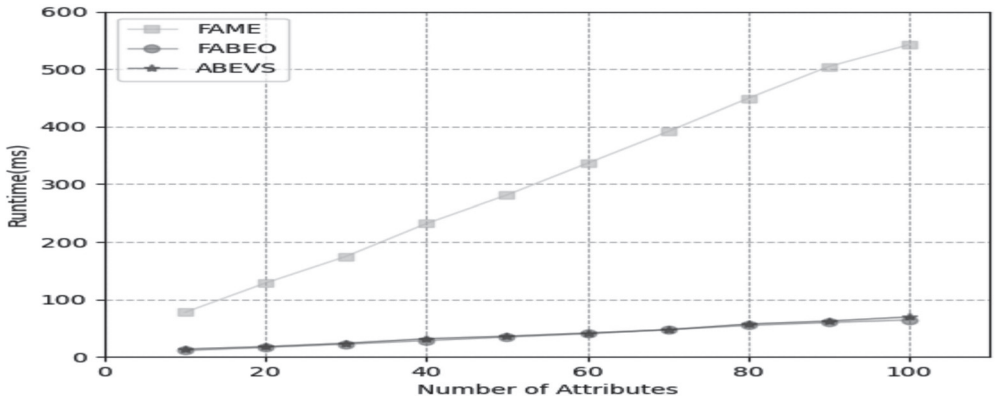

Figure 3 User key generation time.

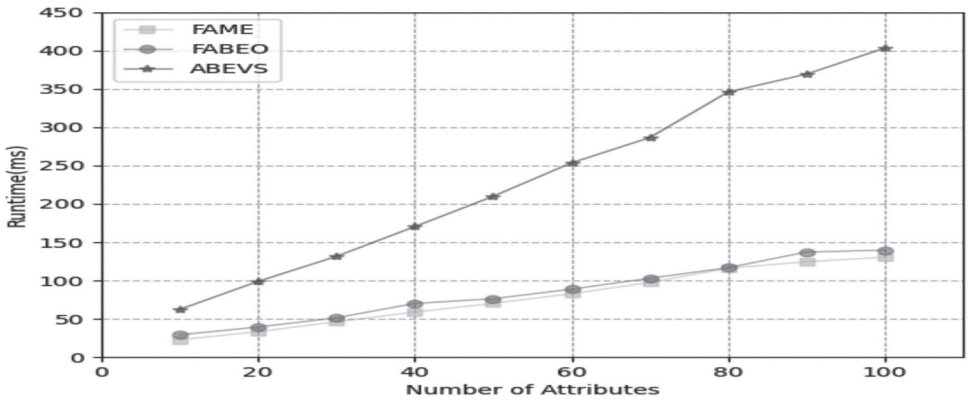

Figure 4 User encryption time.

encryption scheme ABEVS with verifiable signatures used in this paper is adapted from FABEO. Therefore, it is not possible to perform the same function on ABEVS, FABEO [10] and FAME [22]. These three attribute-based encryption schemes are analyzed experimentally, and the experimental operation environment is consistent with the experimental environment of RBACP. As shown in Fig. 3, in terms of key generation, the running time of all three schemes is linearly related to attributes. As shown in Fig. 4, in the user encryption phase, the running time of the three schemes is also linearly related to the attributes and, since ABEVS has the verification function, there is a signature operation in the encryption phase, which leads to the running time of ABEVS being slightly higher than that of FABEO. As shown in Fig. 5, in the user decryption phase, the running time of the three schemes is independent of the attribute set, and it is a relatively smooth state. Moreover, the decryption running time of the proposed scheme is lower than FAME, while the running time of this scheme is higher than FABEO because it needs to verify the ciphertext in the decryption phase.

To summarize, ABEVS has the advantages of high efficiency of FABEO and the function of verifying ciphertext, which is more practical. The RBACP obtained by combining ABEVS with rewritable blockchain improves the efficiency

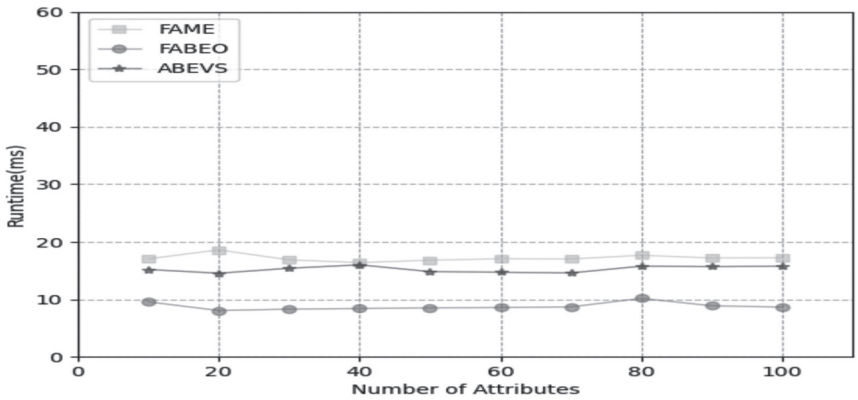

Figure 5 User decryption time.

of uploading and modifying data to a great extent, and the verifiability that ABEVS has greatly improved the security of the blockchain system. Compared with other schemes, the scheme proposed in this paper ensures the efficiency of power data sharing while guaranteeing data security, which is particularly suitable for power data terminal devices with limited resources.

7. CONCLUSION

In this paper, we explore a new approach to power data management in smart grids by proposing an innovative scheme that incorporates rewritable blockchain technology and attribute-based cryptography. In the complex environment of smart grids, it is crucial to ensure data security, transparency, and flexibility. Our solution not only leverages the core benefits of blockchain technology, such as data immutability and decentralization, but also introduces ABE technology for data integrity protection and fine-grained access control. With this combination, we are able to provide a platform that is both secure and flexible enough to support power data management in the smart grid. Future work will further explore and optimize the performance of this solution, including improving data processing efficiency, reducing system operating costs, and extending the applicability of the solution to a wider range of scenarios.

FUNDING

This work is supported by Science and Technology Project Sponsored by State Grid Shanxi Electric Power Co., Ltd. (Research on Autonomous and Controllable Security Gateway and Abnormal Traffic Detection System in Power System, Project No. 5205-302-30008).

REFERENCES

1. CHEN Shuyong, SONG Shufang, LI Lanxin, et al. An overview of smart grid technologies. Power Grid Technology, 2009 (8): 1–7.
2. MA Xuejun, ZHANG Yongshan. Blockchain data privacy protection mechanism for enterprise finance and data mining algorithms. Engineering Intelligent Systems, 2024(32): 435–443.
3. SAHAI A, WATERS B. Fuzzy Identity-Based Encryption. Advances in Cryptology-EUROCRYPT 2005: 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22–26, 2005. Techniques, Aarhus, Denmark, May 22–26, 2005. Proceedings 24. Springer Berlin Heidelberg, 2005: 457–473.
4. YAGA D, MELL P, ROBY N, et al. Blockchain Technology Overview. arxiv preprint arxiv:1906.11078, 2019.
5. TRUONG N B, SUN K, LEE G M, et al. Gdpr-Compliant Personal Data Management: A Blockchain-based Solution. IEEE Transactions on Information Forensics and Security, 2019, 15: 1746–1761.
6. ATENIESE G, MAGRI B, VENTURI D, et al. Redactable Blockchain-or-Rewriting History in Bitcoin and Friends.2017 IEEE European symposium on security and privacy (Euro S&P). IEEE, 2017: 111–126.
7. KRAWCZYK H, RABIN T. Chameleon Hashing and Signatures. Cryptology ePrint Archive, 1998.
8. DERLER D, SAMELIN K, SLAMANIG D, et al. Fine-Grained and Controlled Rewriting in Blockchains: Chameleon-Hashing Gone Attribute-Based. Cryptology ePrint Archive, 2019.
9. CAMENISCH J, DERLER D, KRENN S, et al. Chameleon-Hashes with Ephemeral Trapdoors: and Applications to Invisible Sanitizable Signatures. Public-Key Cryptography-PKC 2017: 20th IACR International Conference on Practice and Theory in Public-Key Cryptography, Amsterdam, The Netherlands, March 28–31, 2017, The Netherlands, March 28–31, 2017, Proceedings, Part II 20. Springer Berlin Heidelberg, 2017: 152–182.
10. RIEPEL D, WEE H. FABEO: Fast Attribute-Based Encryption with Optimal Security. Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. 2022: 2491–2504.
11. TIAN Y, LI N, LI Y, et al. Policy-Based Chameleon Hash for Blockchain Rewriting with Black-Box Accountability. Annual Computer Security Applications Conference. 2020: 813–828.
12. PANWAR G, VISHWANATHAN R, MISRA S. ReTRACe: Revocable and Traceable Blockchain Rewrites Using Attribute-Based Cryptosystems. Proceedings of the 26th ACM Symposium on Access Control Models and Technologies. 2021: 103–114.
13. XU S, NING J, MA J, et al. Revocable Policy-Based Chameleon Hash. Computer Security-ESORICS 2021: 26th European Symposium on Research in Computer Security, Darmstadt, Germany, October 4–8, 2021, Proceedings, Part I 26. Springer International Publishing, 2021: 327–347.

14. MA J, XU S, NING J, et al. Redactable Blockchain in Decentralized Setting. IEEE Transactions on Information Forensics and Security, 2022, 17: 1227–1242.
15. ZHANG D, LE J, LEI X, et al. Secure Redactable Blockchain with Dynamic Support. IEEE Transactions on Dependable and Secure Computing, 2023, 21(2): 717–731.
16. GUO L, LEI H, YAU W C. One-Time Rewritable Blockchain with Traitor Tracing and Bilateral Access Control. 2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom). Security and Privacy in Computing and Communications (TrustCom). IEEE, 2022: 905–912.
17. XU S, NING J, MA J, ET AL. K-Time Modifiable and Epoch-Based Redactable Blockchain. IEEE Transactions on Information Forensics and Security, 2021, 16: 4507–4520.
18. GALBRAITH S D, PATERSON K G, SMART N P. Pairings for Cryptographers. Discrete Applied Mathematics, 2008, 156(16): 3113–3121.
19. AKINYELE J A, GARMAN C, MIERS I, et al. Charm: A Framework for Rapidly Prototyping** Cryptosystems. Journal of Cryptographic Engineering, 2013, 3: 111–128.
20. MIYAJI A, NAKABAYASHI M, TAKANO S. Characterization of Elliptic Curve Traces under FR-Reduction. Information Security and Cryptology-ICISC 2000: Third International Conference Seoul, Korea, December 8–9, 2000 Proceedings 3. Springer Berlin Heidelberg, 2001: 90–108.
21. Zhao P., Jensen A., Johnsen T., "Blockchain Ecosystem Meet Supply Chain Ecosystem and an Application to Dairy Product Provenance", *Engineering Intelligent Systems, 2024*(32): 19–23.
22. AGRAWAL S, CHASE M. FAME: Fast Attribute-Based Message Encryption. Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. 2017: 665–682.