A Compact Ciphertext-Policy Attribute-Based Encryption Scheme for the Information-Centric Internet of Things

JING WANG1,2, NEAL NAIXUE XIONG3, JINHAI WANG4, and WEI-CHANG YEH5.

1School of Data and Computer Science, Sun Yat-Sen University, Guangzhou, China (e-mail: wjing@whu.edu.cn, wangj478@mail.sysu.edu.cn)
2School of Computer Science and Information Security, Guilin University Of Electronic Technology, Guilin, China
3Department of Mathematics and Computer Science, Northeastern State University, Tahlequah, USA (e-mail: xiongnaisxue@gmail.com, xiong31@nsuok.edu)
4College of Electronic and Information Engineering, Foshan University, Foshan, China (e-mail: wangjinhai@whu.edu.cn)
5Department of Industrial Engineering and Engineering Management, National Tsing Hua University, Taiwan (e-mail: weichang.yeh@gmail.com)

Corresponding author: Jing Wang (e-mail: wangj478@mail.sysu.edu.cn).

This work was supported in part by the National Natural Science Foundation of China (Grant No. 61802083).

ABSTRACT The information-centric Internet of things (IC-IoT) is different from the traditional Internet of things (IoT) in that the device-to-device pattern is generalized to a device-to-network pattern. Furthermore, in an IC-IoT environment, there is a demand for protecting the security of all data generated from IC-IoT devices. A cryptography scheme named attribute-based encryption (ABE) represents a smart method of providing fine-grained access control that can sufficiently protect data security. The most attractive advantage of ABE is its expressive access policy, which makes the access control of data flexible and manageable. However, there is a serious problem caused by such an access policy; it incurs a greater ciphertext redundancy and computational overhead. This implies that the current ABE scheme is hard to implement in the thin client devices of IC-IoT. In this paper, we propose a universalized policy-compacting method via sharing public parts of the policy. Compared with the original policy, the compacted policy applies a more compact ciphertext and requires less computation, communication, and storage cost. However, the policy-compacting problem is proved to be an non-deterministic polynomial complete (NPC) problem. Thus, a greedy algorithm is provided to obtain an approximate minimum compacted policy scale. Finally, we propose a compact ciphertext-policy attribute-based encryption (CCP-ABE) scheme with the policy-compacting method. A security proof and performance evaluation show that the proposed CCP-ABE scheme provides a comprehensive performance improvement.

INDEX TERMS information-centric internet of things (IC-IoT), access control, attribute-based encryption (ABE), policy compacting

I. INTRODUCTION

The rise of the information-centric Internet of things (IC-IoT) indicates a pattern change of information exchange and content sharing. Furthermore, the data security demand of IC-IoT is different from traditional IoT [1], [2]. It requires protecting the privacy of the shared content in an IC-IoT network [3]–[5]. Data access control is an effective way to support secure data sharing. The traditional access control mechanism requires a delegation administrator to manage access privilege, and the security of such a mechanism depends entirely on the administrator. However, the highly pervasive and distributed IC-IoT environment presses for a more scalable and flexible access control mechanism [6]. Fortunately, attribute-based encryption (ABE), as a security cryptosystem, can provide fine-grained ciphertext access control for IC-IoT. Different from other kinds of cryptography, such as symmetric and asymmetric cryptography, ABE supports a one-to-many encryption pattern. This implies that a ciphertext of ABE can be decrypted by a set of different secret keys, which improves the scalability and flexibility of ABE. In brief, the core properties of an ABE ciphertext access control mechanism are: (1) the access control of data...
is maintained by a data owner (i.e., a device of the IC-IoT) instead of the storage service provider (i.e., another device of the IC-IoT); (2) the access privilege of users (i.e., all devices of IC-IoT) is described by an access policy, which is more intuitive and readable; and (3) the security property of ABE is derived from cryptography. Therefore, the ABE ciphertext access control mechanism provides several attractive advantages over other access control mechanisms, and it is more suitable for IC-IoT.

The most attractive advantage of ABE is its flexible and expressive access policy, which is used to describe fine-grained data access privilege. Furthermore, the access policy can be managed in a scalable way [33]. However, the complex policy also incurs large ciphertext redundancy. This implies that the large-scale ciphertext of ABE always results in a large overhead: (1) a large computational overhead during encryption and decryption; (2) a high communication overhead during ciphertext uploading and downloading; and (3) a massive storage overhead. As is well known, in IC-IoT, there are a lot of thin client devices with limited resources, and the overhead of ABE is too heavy for these devices. Thus, an effective way of reducing ciphertext redundancy is necessary to improve the existing ABE scheme.

The most significant challenge of low-overhead ABE research is reducing ciphertext redundancy without sacrificing additional performance. In order to reduce ciphertext redundancy, Herranz et al. [17] and Chen et al. [18] proposed constant ciphertext-policy ABE (CP-ABE) schemes, but these schemes were only provided with an expression-limited policy (i.e., AND gate access structure or a threshold function). In order to reduce the computation of client cost by large-scale ciphertext, Hohenberger et al. [25] and Lai et al. [26] provided outsourced computation ABE schemes. In these schemes, most of the computation of encryption or decryption is outsourced to the service providers of the network. Thus, such schemes have a high communication overhead. In order to reduce the policy scale, Zhou et al. [28] and Song et al. [29] provided a minimum sum of product expression (minimum SOPE) and minimum linear code, respectively, to minimize the policy scale. However, although a small-scale policy has less ciphertext redundancy, the reduction of redundancy is limited and unstable. Thus, it is hard to propose an ideal method to reduce the ciphertext redundancy of ABE without sacrificing performance.

In this work, we propose a compact ciphertext-policy ABE (CCP-ABE) scheme to compact the policy scale and reduce ciphertext redundancy. As is well known, there are two kinds of ciphertexts in the CP-ABE scheme: data ciphertext and attribute ciphertext, which are associated with data and attributes, respectively. Furthermore, the number of attribute ciphertexts increases with the scale of the access policy. Thus, policy-compacting, which decreases the policy scale, is an effective way to reduce ciphertext redundancy. In our proposed CCP-ABE, all attribute ciphertexts are divided into two categories: public and private attribute ciphertext units. Different from the private unit, the public unit is shared by multiple parties of the access policy. This implies that multiple private units can be merged as one public unit, and the multiple policy parties associated with the same public unit can be compacted as one. As a result, ciphertext redundancy is reduced by merging ciphertext and compacting policy. Although a cross-utilization public unit could trigger a risk of data leak, our CCP-ABE scheme provides effective protection to avoid such a risk, with only little additional storage overhead.

The main contributions of this paper are summarized as follows:

1. We propose a CCP-ABE scheme to reduce the ciphertext redundancy by sharing public parties of the access policy and public attribute ciphertext units.

2. Two metrics, the flexible factor and overlap factor, are provided to evaluate the policy-compacting efficiency and compact ratio. Thus, the reduction of ciphertext redundancy is more intuitional and measurable.

3. The policy-compacting problem is proven to be a non-deterministic polynomial complete (NPC) problem, and thus, a greedy compacting algorithm is provided to obtain the approximate minimum compact-policy and ciphertext scales.

The remaining of this paper is organized as follows. Section II gives the related work. Then, we propose the policy compacting method in Section III and present the CCP-ABE scheme in Section IV, respectively. Thirdly, we analyse the performance of the proposed compacted policy and CCP-ABE scheme in Section V. Finally, the conclusions are given in Section VI.

II. RELATED WORK

Substantial changes have occurred in information technology (IT), which have also brought various challenges to information security [9]–[11]. For instance, IC-IoT provides a novel data sharing method which is different from traditional IoT. However, data security and privacy become critical issues that restrict IC-IoT development [12], [13]. This is because, in the open access environment, it is hard for the data owner to prevent sensitive data leakage, which incurs a serious security risk to IC-IoT [14]–[16].

An access control mechanism provides an efficient way to protect the data security of IC-IoT. Significantly, scalability and flexibility are two important properties for an IC-IoT data access control mechanism [6]. In the traditional access control mechanism, there is a central organization responsible for managing data access privilege [6]–[8]. This implies that such mechanisms are always restricted in scalability. Fortunately, ABE provides a novel data access control mechanism, where its security depends on cryptography instead of a central privileged organization. Thus, its scalability is effectively improved. Furthermore, ABE provides a novel one-to-many encrypting pattern, which is different from other cryptographies [19]. It is well-suited for the device-to-network pattern in the IC-IoT network. Specifically, ABE can provide a fine-grained ciphertext access control mechanism for IC-IoT by using an expressive access policy. Such an
access policy results in high flexibility of the ABE access control mechanism. Furthermore, in a CP-ABE scheme, the access privilege of data is described by an access policy which is derived from a secret sharing scheme [34]. A wide range of studies have been conducted to design secret sharing schemes, such as the schemes proposed in references [35]–[43]. The access policy is often expressed in various forms, such as the monotonic Boolean formula, an access tree, or a linear secret sharing scheme (LSSS). In brief, the access policy of ABE is diverse, which makes the ABE access control mechanism more scalable and flexible [33].

However, the access policy incurs larger costs of computation, communication, and storage, which limits its commercial applications [30]–[32]. Thus, many researchers focus on the low-cost ABE schemes and low-cost applications of ABE [20]–[22]. As shown in Table 1, there are three ways to achieve low-cost ABE schemes, as recently reported: constant-ciphertext setting, computation outsourcing, and policy minimizing. In a constant-ciphertext ABE scheme [23], [24], the access policy is expressed as an AND gate or a threshold function. Although this simple access policy reduces the resource costs of clients, it is also limited in the expression. In order to reduce client computation cost, some ABE schemes provide outsourcing of the decryption function [25]–[27]. Although the computation cost of the client is reduced, the communication overhead is increased. The other efficient way to reduce resources of the ABE scheme is by minimizing the access policy. Minimal sum-of-product expression (minimum SOPE) [28] and minimum linear code [29] schemes are provided to minimize policy size without breaking policy logic. This implies that the system overhead and policy performance of these schemes are all optimized. It seems that minimum policy ABE is optimal, as the system overheads are all reduced and the access policy is strong at expression. However, the performance of minimum policy ABE is limited and unstable.

Considering the insufficiencies of the above low-cost ABE schemes, we propose CCP-ABE, which has the advantage of comprehensive performance. It provides an efficient and stable way to reduce the ciphertext redundancy of ABE without any additional restrictions or costs. Furthermore, the policy-compacting method can also be universally used to reduce the overhead of various existing ABE schemes.

### III. COMPACTED POLICY FOR CIPHERTEXT-POLICY ATTRIBUTE-BASED ENCRYPTION

ABE can be viewed as a tuple \( \{ E_M, E_A, A, Share \} \), where \( E_M \) and \( E_A \) are encryption algorithms, \( A \) denotes the attribute set, and \( Share \) is a secret sharing scheme (SSS). In most CP-ABE schemes, plaintext \( M \) is encrypted with a random secret \( s \) as \( C_0 = E_M(M, s) \). Then, \( s \) is divided into a set of shares \( S = \{ s_1, s_2, \ldots, s_n \} \) by \( Share \). Finally, each share is encrypted with an attribute public key (PK) as \( C_1 = E_A(s_i, PK_{\rho(i)}) \), where \( \rho \) denotes a map from the labels \( \{1, 2, \ldots, n\} \) to \( A \), and \( PK_{\rho(i)} \) denotes the PK of \( \rho(i) \in A \). Thus, the whole ciphertext is expressed as \( CT = \{ C_0, C_1, \ldots, C_n \} \). Significantly, the size of \( CT \) is linearly increased with the size of \( S \).

Assume that a data owner encrypts a set of data and generates multiple attribute ciphertext units associated with the same attribute. If these units are also assigned with the same share, they can be compacted as one unit, which is called the public attribute ciphertext in this work. In this vein, we present a method to compact the share set and reduce the ciphertext redundancy of ABE.

### A. ACCESS POLICY

The access policy, also called the access structure, is a core concept of ABE. The formal definition of the access structure is given as follows.

**Definition 1 (Access Structure [19]):** Let \( \{ P_1, P_2, \ldots, P_n \} \) be a set of parties\(^1\). An access structure is a collection \( A \) of non-empty subsets of \( \{ P_1, P_2, \ldots, P_n \} \). The sets in \( A \) are called the authorized set, and the sets not in \( A \) are called the unauthorized sets. Furthermore, an access structure \( A \) is monotonic if \( \forall B, C: B \in A \land B \subseteq C \rightarrow C \in A \).

There are three common access policy forms: access tree, LSSS matrix, and monotonous Boolean expression [33]. For simplicity, we only discuss the access tree in this work. An

\(^1\)In the ABE context, the role of the parties is taken by the attributes.

### TABLE 1: Comparison of Existing Low-Cost Schemes.

<table>
<thead>
<tr>
<th>Scheme</th>
<th>Computation Overhead</th>
<th>Communication Overhead</th>
<th>Storage Overhead</th>
<th>Access Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant-Ciphertext</td>
<td>reduced</td>
<td>reduced</td>
<td>reduced</td>
<td>AND Gate, limited</td>
</tr>
<tr>
<td>Threshold CP-ABE</td>
<td>reduced</td>
<td>reduced</td>
<td>reduced</td>
<td>Threshold function, limited</td>
</tr>
<tr>
<td>Computation Outsourcing</td>
<td>reduced</td>
<td>increased</td>
<td>-</td>
<td>LSSS(^2), monotone</td>
</tr>
<tr>
<td>Outsourced Decryption</td>
<td>reduced</td>
<td>increased</td>
<td>-</td>
<td>LSSS, monotone</td>
</tr>
<tr>
<td>DAC-MACS(^2)</td>
<td>reduced</td>
<td>reduced</td>
<td>reduced</td>
<td>Boolean formula, monotone</td>
</tr>
<tr>
<td>Minimum SOPE(^3)</td>
<td>reduced</td>
<td>reduced</td>
<td>reduced</td>
<td>LSSS, monotone</td>
</tr>
<tr>
<td>Minimum linear code(^4)</td>
<td>reduced</td>
<td>reduced</td>
<td>reduced</td>
<td>access tree/LSSS/Boolean formula(^5), monotone</td>
</tr>
</tbody>
</table>

---

1. ABOE: Attribute-Based Online/Offline Encryption.
2. DAC-MACS: Effective Data Access Control for Multi-authority Cloud Storage Systems
4. LSSS: Linear Secret Sharing Scheme.
5. For simplicity, only the computed access tree is given in this work. However, other compacted policy forms can be obtained by the policy transforming method of reference [33].
access tree is special kind of tree structure. Let $T$ be an access tree. Each non-leaf node $n$ of $T$ is a $t_n$-out-of-$n_n$ node (i.e., threshold), where $n_n$ denotes the number of its children, $t_n$ denotes its threshold, and $0 \leq t_n \leq n_n$. Each leaf $n$ of $T$ is described by an attribute $\rho(n)$ and the threshold $t_n = 1$. Then, according to Shamir’s secret sharing scheme [35], each node $n$ is assigned with a node polynomial $f_n(x)$, such that:

$$f_n(0) = \begin{cases} s, & n \text{ is the root of } T, \\ p, & \text{Otherwise} \end{cases}$$

(1)

$$d_n = t_n - 1,$$  

(2)

where $s$ denotes the secret, $p$ denotes the parent of $n$, $x_n$ denotes the interpolation of $n$, and $d_n$ denotes the degree of $f_n(x)$. Finally, $s_n = f_n(0)$ is called the node share of $n$.

Furthermore, the secret $s$ (i.e., node share of the root of $T$) can be reconstructed by an authorized attribute set $A$. The reconstruction is processed recursively as follows. If $n$ is a leaf, $s_n$ can be reconstructed if and only if $\rho(n) \in A$. Otherwise, $n$ is a non-leaf, and the reconstruction of $s_n$ is successful if and only if at least $t_n$ node shares of its children are successfully reconstructed. Let $N_n$ be a $t_n$-sized child set of $n$, where all $c_i \in N_n$ have successfully been reconstructed with node share $s_{c_i}$. The share of $n$ can be calculated as follows:

$$s_n = \sum_{c_i \in N_n} s_{c_i} \Delta_{i,X_n}(0),$$  

(3)

$$\Delta_{i,X_n}(x) = \prod_{j \in X_n, j \neq i} \frac{x - j}{i - j},$$  

(4)

where $f_n(i) = s_{c_i}$, $X_n = \{i | c_i \in N_n\}$, and $\Delta_{i,X_n}(x)$ is called the Lagrange coefficient. The correctness proof is shown as follows:

$$\sum_{c_i \in N_n} s_{c_i} \Delta_{i,X_n}(0) = \sum_{c_i \in N_n} f_n(i) \Delta_{i,X_n}(0) = f_n(0) = s_n,$$  

(5)

### B. POLICY-COMPACTING PROBLEM

As is well known, a node of an access policy can be describe by a attribute and a share. Thus, multiple leaves can be compacted to one when such leaves are assigned with the same attribute and share. Figure 1 shows two examples of policy compacting. Multiple ordinary leaves of access trees can be compacted as one public leaf. In the single policy case, there are two leaves assigned with the attribute $a_2$. The two leaves can be compacted as one public leaf when $s_2$ equals $s_3$. Similarly, in the multi-policy case, four leaves of two policies can be compacted as two public leaves when $s_2 = s_4$ and $s_3 = s_5$. Furthermore, the scale of ABE ciphertexts is dependent on the number of leaf nodes. This implies that the multiple private attribute ciphertext units associated with the ordinary leaves can also be compacted as one public attribute ciphertext unit associated with the public leaf. Thus, the ciphertext scale can be effectively reduced by using the compacted access policies and the public ciphertext units.

However, there may be a risk of information leaking incurred by public ciphertext units. Suppose that an owner uploads his data $M_1$ and $M_2$, as shown in Figure 2. Let $s_1$ and $s_2$ be the secrets of $M_1$ and $M_2$, respectively; $T_1$ and $T_2$ are the access trees assigned to $M_1$ and $M_2$, respectively; and a user gets the attribute set $S_u = \{a_1, a_2, a_3\}$. It is clear that the user is not allowed to access $M_2$, because $S_u$ is an unauthorized set of $T_2$. However, in this case, the user can illegally access $M_2$ as follows. First, they recover shares $s_1^*, s_2^*$, and $s_4^*$ via $S_u$. Then, they calculate:

$$s_3^* = \frac{x_2 - x_3}{x_2 - x_1} s_1^* + \frac{x_1 - x_3}{x_1 - x_2} s_2^*.$$  

(6)

Finally,

$$s_2 = \frac{x_5}{x_5 - x_4} s_4^* + \frac{x_4}{x_4 - x_5} s_3^*.$$  

(7)

As a result, they can successfully recover $M_2$ via the unauthorized $S_u$. In order to prevent the information from leaking, the interpolations $x_3$ and $x_4$ of public unit $C_{a_3}^*$ must be
encoded by the associated attribute \(a_3\). Thus, the user cannot get \(x_3\) and \(x_4\) unless the attribute \(a_3 \in S_u\). Furthermore, \(s_2\) cannot be calculated by Equation (6), \(s_2\) cannot be recovered by Equation (7), and \(M_2\) cannot be illegally accessed.

The other challenge of policy compacting is constructing the optimal compacted policies, which correspond to the minimum ciphertext scale. The formal definition of the optimal policy-compacting problem is given as follows:

Input: \((\mathcal{T}, k)\), where \(\mathcal{T}\) is an access policy set, and \(k \in \mathbb{Z}^+\). Question: Does \(\mathcal{T}\) have a valid share set \(\mathcal{S}\) with size \(k\)?

Claim 1: The problem is non-deterministic polynomial (NP) hard.

Proof:
Let \(\pi\) be a function that assigns node shares for the access tree. \(\mathcal{L}\) be a set, and element \((a, s) \in \mathcal{L}\) be described by an attribute \(a\) and a share \(s\). The following verifier of the problem runs in polynomial time of \(|\mathcal{L}|\):

Verifier \(V((\mathcal{T}, k), (\pi, \mathcal{L}))\).

The verifier output is true if and only if all the following conditions are true:

- \(|\mathcal{L}| < k\)
- \(\forall n \in \mathcal{T} \text{ and } \mathcal{T} \in \mathcal{T}, \pi(n)\) must be calculated efficiently.
- \(\forall \text{ leaf } n \in \mathcal{T} \text{ and } \pi(n) \in \mathcal{L}, \rho(n)\) denotes the attribute associated with \(n\).

Claim 2: 3-satisfiability(3-SAT) \(\leq_p\) policy compacting.

Proof:
Define a function \(f\) with input \(\varphi\) and outputs \((\mathcal{T}, k)\), where \(\varphi\) is an instance of 3-SAT and \((\mathcal{T}, k)\) is an instance of policy-compacting. We now show that \(f\) is a polynomial-time function which converts the policy-compacting problem into a 3-SAT problem.

Let \(X = \{x_1, \ldots, x_n\}\) denote the literals of \(\varphi\), and \(C = \{c_1, \ldots, c_m\}\) denote the clauses of \(\varphi\). To justify this claim, suppose \(k = 2n + 3\) and \(\mathcal{A} = \{y_1, z_1, \ldots, y_n, z_n\} \cup \{\omega, \bar{\omega}\}\) is the attribute set, where \(y_i\) represents \(x_i\), \(z_i\) represents \(\neg x_i\), and \(\omega, \bar{\omega}\) denote the adding attributes.

FIGURE 3: Clauses \(c_j\) transform into access policy \(\bar{c}_j\).

\((\wedge: \text{AND gate}; \vee: \text{OR gate}; \alpha, \beta, \gamma, \in X; \omega, \bar{\omega}: \text{adding attributes})\)

For each \(c_j = \alpha \vee \beta \vee \gamma \in C\), we get an access tree \(\bar{c}_j\), as shown in Figure 3. Thus, \(\mathcal{T} = \{c_1, \ldots, c_m\}\). Let \(F \in \{0, 1\}^n\) denote an assignment of literal set \(X\), and let \(F_i\) denote the value of \(x_i\). If \(\varphi\) is satisfied, \(\exists F\) that makes \(\varphi\) true. We choose \(a, b \in Z\) at random and define a function as follows:

\[
\tau(y_i) = \begin{cases} 
  a, & F_i = 1 \\
  b, & \text{otherwise} 
\end{cases}
\]

\[
\tau(z_i) = \begin{cases} 
  b, & F_i = 1 \\
  a, & \text{otherwise} 
\end{cases}
\]

FIGURE 4: Choosing node shares of \(\bar{c}_j\).

\((s: \text{the secret}; f_1(x): \text{node polynomial}; \tau(*): \text{assignment function}; \wedge: \text{AND gate}; \vee: \text{OR gate}; \alpha, \beta, \gamma, \in X; \neg *=: \text{negation of } *=; \omega, \bar{\omega}: \text{adding attributes})\)

Let \(f_1(x), f_2(x), f_3(x), \text{ and } f_4(x)\) be four non-constant functions assigned as node polynomials of \(\bar{c}_j\). Then, an interpolation set \(S_x = \{x_1, x_\alpha, x_\beta, x_\gamma\} \in Z_p^*\) is chosen at random, and \(f_1(x)\) is subject to:

\[
\begin{align*}
  f_1(x_1) &= b \\
  f_1(x_\alpha) &= \tau(\alpha) \\
  f_1(x_\beta) &= \tau(\beta) \\
  f_1(x_\gamma) &= \tau(\gamma)
\end{align*}
\]

Then, \(f_1(x)\) can be constructed as follows:

\[
f_1(x) = b\Delta_{S_x,x_1}(x) + \sum_{i \in \{\alpha, \beta, \gamma\}} \tau(i)\Delta_{S_x,x_i}(x).
\]

Let \(f_2(x) = c_2x + s\), where \(c_2 \in Z_p\) and \(s = f_1(0)\). There are two solutions, \(x_{2a} = (a - s)/c_2 \in Z_p\) and \(x_{2b} = (b - s)/c_2 \in Z_p\), of the equations \(f_2(x_a) = a\) and \(f_2(x_b) = b\), respectively. The node polynomials \(f_3(x)\) and \(f_4(x)\) are constructed in the same manner. The node shares of \(\bar{c}_j\) are chosen as shown in Figure 4, and at most nine elements are added to \(\mathcal{L}\):

\[
\mathcal{L} \leftarrow \mathcal{L} \cup \{(\alpha, \tau(\alpha)), (\neg \alpha, \tau(\neg \alpha)), (\beta, \tau(\beta)), (\neg \beta, \tau(\neg \beta)), (\gamma, \tau(\gamma)), (\neg \gamma, \tau(\neg \gamma)), (\omega, \tau(\omega)), (b, \tau(\bar{\omega})), (\bar{\omega}, \tau(b))\}.
\]

There must be three solutions of the equation \(f_3(x) = b\), otherwise the third-order polynomial function \(f_3(x)\) degenerates to a constant function \(\bar{f}(x) = b\). Thus, the node shares shown in Figure 4 are valid if and only if \(\bar{c}_j\) is true (i.e., at most two of the variables \(\tau(\alpha), \tau(\beta), \tau(\gamma)\) are set to be \(b\)). Furthermore, all \(\bar{c}_j\), \(1 \leq j \leq m\) can be assigned node shares as shown in Figure 4 if and only if \(F\) makes \(\varphi\) true. Finally, we find:

\[
\mathcal{L} = \{\omega, \alpha, \langle \omega, b \rangle, \langle \bar{\omega}, b \rangle, \langle y_1, \tau(y_1) \rangle, \langle z_i, \tau(z_i) \rangle: 1 \leq i \leq n\}.
\]

Thus, \(|\mathcal{L}| = 2n + 3\) in this case.

If \(\varphi\) is unsatisfied, \(\forall F, \exists c_{k_1} = \alpha_{k_1} \vee \beta_{k_1} \vee \gamma_{k_1}\) and \(c_{k_2} = \alpha_{k_2} \vee \beta_{k_2} \vee \gamma_{k_2}\), where \(\alpha_{k_1}, \beta_{k_1}, \gamma_{k_1}\) are all false and \(\alpha_{k_2}, \beta_{k_2}, \gamma_{k_2}\) are all true. Because \(\varphi\) is unsatisfied, \(\exists c_{k_1}\) that is unsatisfied (which implies that \(\alpha_{k_1}, \beta_{k_1}, \gamma_{k_1}\) are all false).
If $\neg \beta_{k_2}$, where $\alpha_{k_2}, \beta_{k_2}, \gamma_{k_2}$ are all true, then this implies $\forall c_j$ there is at least one false literal. Thus, $\overline{F}$ (the negation of $F$) is a satisfied assignment for $\varphi$.

**Definition 2 (Flexibility factor $\gamma_n$ for node $n$):** The flexibility factor $\gamma_n$ denotes the total number of nodes in $N_n$ that can be assigned with random shares.

The upper limit of $\gamma_n$ is given as follows:

$$\gamma_n = \begin{cases} t_n, & t_n \neq 2 \\ |N_n|, & t_n = 2 \end{cases}. \quad (12)$$

Following the definition, for all subtrees $T' \subset T$, the following inequality must hold:

$$\sum_{n \in T'} \gamma_n \leq | \bigcup_{n \in T'} N_n |. \quad (13)$$

Thus, $\gamma_n = \min \{ \gamma_n \leq |N_n| \}$, where $N_n = \{ c_i \gamma_{c_i} = |N_{c_i} \cap N_n|, 1 \leq i \leq n \}$. The detailed proof of this equation is given in Appendix A.

Similarly, the overlap factor is defined as follows.

**Definition 3 (Overlap factor $\delta_T$ for access tree $T$):** The overlap factor $\delta_T$ of $T$ denotes the total number of leaves of $T$ that can be associated with public units.

We then get:

$$\delta_T = \sum_{n \in N} \gamma_n - 1. \quad (14)$$

The validity of Equation (14) is proved in Appendix B.

Furthermore, the bilinear map plays a crucial role in ABE. The definition of a bilinear map is described as follows.

**Definition 4 (Bilinear Map):** Assume that $G_0, G_T$ are two multiplicative cyclic groups with prime order $p$, and $g$ is a generator of $G_0$. Function $e : G_0 \times G_0 \rightarrow G_T$ is a bilinear map if and only if it satisfies three criteria:

1) **Bilinearity:** $\forall u, v \in G_0$ and $a, b \in \mathbb{Z}_p$, $e(u^a, v^b) = e(u, v)^{ab}$;
2) **Non-degeneracy:** $\forall u, v \neq 0, e(u, v) \neq 1$;
3) **Computability:** $e$ must be computed efficiently.

Additionally, there are some hard problems which support the security of the ABE mechanism. We introduce one of the hard problems—the decisional $q$-parallel bilinear Diffie-Hellman exponent assumption—to guarantee the security of our CCP-ABE scheme.

**Assumption 1 (Decisional $q$-Parallel Bilinear Diffie-Hellman Exponent Assumption, $q$-parallel BDHE):** Assume $G_0, G_T$ are two group with prime order $p$, and $e : G_0 \times G_0 \rightarrow G_T$ is a bilinear map. Let $\alpha_1, \ldots, \alpha_q, \beta_1, \ldots, \beta_q, s \in \mathbb{Z}_p$ be chosen at random, and let $g$ be a generator of $G_0$. If a probabilistic polynomial-time (PPT) adversary $A$ is given:

$$\overline{y} = \{ g, g^s, g^{\alpha_1}, \ldots, g^{\alpha_q}, g^{\alpha_{q+1}}g^{\alpha_q}, \ldots, g^{q2g};$$

then it must be hard to distinguish a valid element $T_0 = e(g, g)^{a1+1}$ from a random element $T_1 = R \in G_T$. Assume that $B$ is a PPT algorithm with output $z \in \{0, 1\}$. We say that $B$ gets the advantage $\epsilon$ in solving $q$-parallel BDHE, if:

$$|Pr[B(\overline{y}, T = T_0)] - Pr[B(\overline{y}, T = T_1)]| \geq \epsilon. \quad (15)$$
where \(Pr[*]\) denotes the probability of event \(*\). The decisional \(q\)-parallel BDHE assumption holds if and only if there is no PPT algorithm \(B\) that gets a non-negligible advantage \(\epsilon\) in distinguishing the \(q\)-parallel BDHE tuple \(\{\tilde{y}, T\}\).

**D. GREEDY POLICY-COMPACTING ALGORITHM**

**Algorithm 1 Init\(_{\gamma}(n)\)**

**Require:** node: \(n\)

**Ensure:** State flag: \(\mu_n \in \{0, 1\}\)

1: if \(n\) is a leaf then
2: return 0
3: else
4: //\(n\) is a \(t_n\)-out-of-\(n_n\) node and \(N_n\) be the child set of \(n\)
5: \(k_n \leftarrow 0\)
6: for \(c_i \in N_n - \{n\}\) do
7: \(k_n \leftarrow k_n + Init(c_i)\)
8: end for
9: if \(t_n = 2\) then
10: \(\gamma_n \leftarrow \min\{n_n, |N_n| - k_n\}\)
11: else
12: \(\gamma_n \leftarrow \min\{t_n, |N_n| - k_n\}\)
13: end if
14: if \(\gamma_n = |N_n| - k_n\) then
15: return 1
16: else
17: return 0
18: end if
19: end if

Before compacting access polices, we need to initialize the flexibility factor of each node via Algorithm 1. Let \(T\) be an access tree, and let \(r\) be the root of \(T\). \(Init_r(r)\) is run as a depth-first traversal of \(T\) and initializes the flexibility factor of all nodes of \(T\).

Then, the algorithm \(Update(n)\) is called to update the flexibility factor of the access tree \(T\) when a node \(n \in T\) is assigned with node share. In this algorithm, three arrays \(Count, C, S\) are given to describe the public attribute ciphertext. For each attribute \(a_i\), there are three related parameters \(count_i, C_n^{a_i} \in C, s_i^n \in S\). \(count_i\) denotes the number of leaves \(n \in T\) associated with \(a_i\), \(C_n^{a_i}\) denotes the public ciphertext unit of \(a_i\), and \(s_i^n\) denotes the public share of \(a_i\). Clearly, \(C_n^{a_i}\) and \(s_i^n\) are all initialized to be \(null\) at the beginning. Then, \(count_i, C_n^{a_i}, s_i^n \in S\) are all updated when a leaf \(n \in T\) with attribute \(a_i\) is assigned with a node share \(s_n\). Finally, \(Update(n)\) is called iteratively to assign the share for \(n\) and update the flexibility factor of part of the node in \(T\).

Suppose that \(T = \{T_1, T_2, \ldots, T_m\}\) denotes an access tree set, and \(Compact(T)\) is a greedy algorithm proposed for compact-policy set \(T\). First, array \(Count\) is initialized. Then, \(Compact(T)\) calls \(Init_{\gamma}(r_j)\) for each root \(r_j \in T\) to initialize its flexibility factor. Third, each secret \(s_j\) of \(r_j\) is assigned, and \(Update_{\gamma}(r_j)\) is called to update the flexibility factor again. Finally, the attribute \(a_i\) with max \(count_i\) is chosen each time, and a share \(s_i^n\) is assigned for all nodes \(n\) where \(\rho(n) = a_i\) and \(s_n = null\). In this step, for all leaves \(n, \rho(n) = a_i\), \(Update_{\gamma}(n)\) is called to update their flexibility factor again.

**Algorithm 2 Update\(_{\gamma}(n)\)**

**Require:** Node: \(n\)

**Ensure:** Node share: \(s_n\) Updated node flexibility factor: \(\gamma_n\)

1: if \(n\) is a leaf then
2: if \(s^{*}_{\rho(n)} = null\) then
3: if \(s_n = null\) then
4: \(s_n \leftarrow Z_p\)
5: \(\gamma_n \leftarrow \gamma_n - 1\)
6: \(\gamma_p \leftarrow \gamma_p - 1\)
7: Update\(_{\gamma}(p)\)
8: end if
9: \(s^{*}_{\rho(n)} \leftarrow s_n\)
10: \(\phi(n) \leftarrow \&C^{\rho}(n)\)
11: else
12: if \(s_n \neq null\) then
13: \(s_n \leftarrow s^{*}_{\rho(n)}\)
14: \(\phi(n) \leftarrow \&C^{\rho}(n)\)
15: \(\gamma_n \leftarrow \gamma_n - 1\)
16: \(\gamma_p \leftarrow \gamma_p - 1\)
17: Update\(_{\gamma}(p)\)
18: end else
19: assign the storage unit for \(C_n\)
20: \(\phi(n) \leftarrow \&C^{\rho}(n)\)
21: \(Count_{\rho(n)} \leftarrow Count_{\rho(n)} - 1\)
22: end if
23: end if
24: else
25: if \(n\) is root and \(s_n = null\) then
26: \(s_n \leftarrow Z_p\)
27: \(\gamma_n \leftarrow \gamma_n - 1\)
28: end if
29: if \(\gamma_n = 0\) and \(f_n = null\) then
30: //\(n\) be the child set of \(n\)
31: Calculate node polynomial function \(f_n(x)\)
32: Assign interpolation for all \(a_i \in N_n\)
33: if \(s_n = null\) then
34: \(s_n \leftarrow f_n(0)\)
35: \(\gamma_p \leftarrow \gamma_p - 1\)
36: Update\(_{\gamma}(p)\)
37: end if
38: for \(c_i \in N_n\) do
39: if \(s_{c_i} = null\) then
40: \(s_{c_i} \leftarrow f_n(s_{c_i})\)
41: \(\gamma_{c_i} \leftarrow \gamma_{c_i} - 1\)
42: Update\(_{\gamma}(c_i)\)
43: end if
44: end for
45: end if
46: end if
Algorithm 3 Compact(T)

Require: Access policy set: \( T \)
Ensure: Node share set: \( S \)
1: for leaf \( n \in T \) do
2: \[ \text{count}_{\rho(n)} \leftarrow \text{count}_{\rho(n)} + 1 \]
3: end for
4: for \( \mathbb{T}_t \in T \) do
5: \( \text{Init}_{\gamma}(\tau_i) / \mathbb{T}_t \) denotes the root of \( T \)
6: \( \text{Update}_{\gamma}(\tau_i) \)
7: end for
8: while \( \exists a_i, \text{count}_{a_i} \neq 0 \) do
9: \( \text{//Let } N_{a_i} \text{ be the set of leaves assigned with attribute } a_i \)
10: \( \text{while } N_{a_i} \neq \emptyset \) do
11: \( \text{Choose a node } n \in N_{a_i} \)
12: \( \text{Update}_{\gamma}(n) \)
13: \( N_{a_i} \leftarrow N_{a_i} - \{n\} \)
14: end while
15: end while

IV. COMPACT CIPHERTEXT-POLICY ATTRIBUTE-BASED ENCRYPTION SCHEME

As shown in Figure 6, the system model of the compact ciphertext-policy attribute-based encryption (CCP-ABE) scheme consists of four types of entities:

- Authority. The authority is responsible for generating the public key (PK), secret key (SK), and master key (MK).

- Information center. The information center can be viewed as an abstract distributed cluster of the network. It is responsible for data storage in the IC-IoT network.

- Owner. An owner denotes a device of IC-IoT that generates and uploads data to the information center. Note that data uploaded to the information center are all encrypted as ciphertexts by using the PK.

- User. A user denotes a device that downloads ciphertext from the information center and recovers the according plaintext by its SK. Note that a user can also be an owner in this system.

Moreover, the CCP-ABE scheme includes the following four functional modules:

**Setup:** The authority chooses \( g_1, g_2 \in G_0, w, x \in Z_p \) and calculates \( P = g^r_1 \) and \( \gamma = e(g_1, g_2)^w \), where \( G_0, G_T \) are two multiplicative cyclic groups with prime order \( p \), and \( e : G_0 \times G_0 \rightarrow G_T \) denotes a bilinear map. For each attribute \( a_i \in A, 1 \leq i \leq N \), a random number \( a_i \in Z_p \) is chosen and the public key \( P_i = g_1^{a_i} \) is computed. Then, PK and MK are shown as follows:

\[
PK = \{g_1, g_2, \gamma, P, P_i\}_{1 \leq i \leq N},
MK = \{w, x\}.
\]

**KeyGen:** A user sends their attribute set \( U_s \) to the authority to request their SK. First, the authority picks \( u \in Z_p \) randomly. Then, \( D_u' = g_2^{u-x \cdot w}, D_u = g_2^x \), and \( D_i = g_2^{u \cdot a_i} \) are calculated, where \( a_i \in U_s \). Finally, the SK \( SK_{a_i} = \{D_u', D_u, D_i\}_{n \in U_s} \) is sent to the user.

**Encryption:** Let \( M = \{M_1, M_2, \ldots, M_n\} \) be a plaintext set and let \( T = \{\mathbb{T}_1, \mathbb{T}_2, \ldots, \mathbb{T}_n\} \) denote the according access tree set. The owner calls \( \text{Compact}(T) \) to compact access policies and assigns shares. Then, the ciphertext is calculated in three steps. First, for each \( M_j \in M \), the data ciphertext is calculated:

\[
C_{M_j} = \langle C_j = M_j \mathbb{T}^s_j, C_j = g_1^{s_j} >,
\]
where \( s_j \in Z_p \). Second, for each private node \( n \), the private ciphertext unit is calculated as follows:

\[
C_n = \langle \bar{C}_n = g_1^{r_n}, \bar{C}_n = P_1^{r_n} P^s_n >,
\]
where \( r_n, s_n \in Z_p \). Finally, for each public ciphertext \( C_{a_i}^s \), we calculate:

\[
C_{a_i}^s = \langle \bar{C}_i = g_1^{r_{a_i}}, \bar{C}_i = P_1^{r_{a_i}} P^s_{a_i} >,
\]
where \( r_{a_i}, s_{a_i} \in Z_p \). Additionally, for all nodes \( n, \phi(n) = C_{a_i}^s \), the node interpolation \( x_n \) is encoded as \( \bar{x}_n = x_n \mathbb{T}^t_{a_i} \).

**Decryption:** The user determines an authorized set \( U' \subseteq U_s \), where \( U_s \) is their attribute set. For each private leaf node \( n \in U_j \), they calculate:

\[
TK_n = \frac{e(\bar{C}_n, \bar{D}_n)}{e(C_n, D_\rho(n))} = e(g_1, g_2)^{x \cdot s_n},
\]
where \( \rho(n) \in U' \). For each public leaf node \( n \in U_j \), the user calculates:

\[
TK_n = \frac{e(\bar{C}_n^s, \bar{D}_n)}{e(C_n^s, D_\rho(n))} = e(g_1, g_2)^{x \cdot s_n^\phi(n)},
\]

\[
x_n = \frac{\bar{x}_n e(\bar{C}_n, \bar{D}_n)}{e(D_n', C_n^\rho(n)) e(C_n^\rho(n), D_\rho(n))},
\]
where \( \phi(n) = C_{a_i}^s \). Then, all non-leaf nodes are processed as follows. Let \( n \) be a t-out-of-n node, let \( S_n \) be a child set of n,
and |S\alpha| = t. Assume that \forall \mu \in S\alpha, TK\mu is obtained. The user computes TK_{rj} as follows:

\[
TK_n = \prod_{\mu \in S\alpha} TK_{\Delta \mu, S\alpha}(0) = \prod_{\mu \in S\alpha} (e(g_1, g_2)^{ux_\mu \Delta \mu, S\alpha}(0)),
\]

where \Delta \mu, S\alpha(0) is the Lagrange coefficient of \mu. The user recovers the plaintext when TK_{r_j} of the access tree root \( r_j \) is obtained:

\[
C_j = e(C^*_j, D^*_u) TK_{r_j} = e(g_1, g_2)^{ux_{r_j}} = e(g_1, g_2)^{ux_{r_j}} = M_j.
\]

**V. PERFORMANCE ANALYSIS**

**A. SECURITY PROOF**

We prove that the security of our CCP-ABE scheme in the selective security model reduces to the hardness of the q-parallel BDHE assumption. Suppose that there exists a polynomial-time adversary \( \mathcal{A} \) which can attack our scheme in the selective security model with advantage \( \epsilon \). Then, we can build a simulator \( \mathcal{B} \) which distinguishes the q-parallel BDHE tuple \( \{\tilde{g}, T\} \) with advantage \( \epsilon \). The simulation proceeds as follows.

**Init:** First, \( \mathcal{B} \) gets a challenge q-parallel BDHE tuple \( \{\tilde{g}, T\} \). Note that we only consider the simplest case of a compact access policy in this proof; the proof of other cases is similar. Thus, \( \mathcal{A} \) can present a challenge policy set as \( \mathcal{T} = \{T_1, T_2\} \), where \( |L_{T_1}| = l_1, |L_{T_2}| = l_2, \) and \( l_1 + l_2 \leq q \). Let \( n^* \) be the public node of \( T_i \), which only associates with the \( i_1^{th} \) leaf of \( T_1 \) and the \( i_2^{th} \) leaf of \( T_2 \). Following the method proposed in Reference [33], \( T_1 \) and \( T_2 \) can be equivalently expressed as two LSSS matrices denoted by \((X, \rho_1)\) and \((Y, \rho_2)\). Assume that \( X = (x_{i,j})_{l_1 \times m_1}, Y = (y_{i,j})_{l_2 \times m_2}, X_{i,j}, Y_{i,j} \) denote the rows assigned to public node \( n^* \), \( \rho_1(i_1) = \rho_2(i_2) = a^* \), and \( m_1 \leq l_1, m_2 \leq l_2 \). Following the definition of \( X \) and \( Y \), there exist \( j_1 \in \mathbb{Z}_{m_1}, j_2 \in \mathbb{Z}_{m_2} \) where \( \forall i \neq i_1, x_{i,j_1} \neq 0 \) and \( \forall i \neq i_2, y_{i,j_2} \neq 0 \). For simplicity, let \( j_1 = j_2 = 1 \). Then, we define the following matrix \( M \), and mapping function \( \rho^* \):

\[
M = (m_{i,j})(l_1 + l_2) \times (m_1 + m_2) = \begin{pmatrix}
(x_{1,1})_{l_1 \times m_1} & \cdots & (x_{l_1,1})_{l_1 \times m_1} \\
\vdots & \ddots & \vdots \\
(x_{1,l_1})_{l_1 \times m_1} & \cdots & (x_{l_1,l_1})_{l_1 \times m_1} \\
(x_{i_1,1})_{l_1 \times m_1} & \cdots & (x_{i_1,l_1})_{l_1 \times m_1} \\
(0)_{m_2 \times 1} & \cdots & (0)_{m_2 \times 1} \\
\vdots & \ddots & \vdots \\
(x_{i_1,l_1})_{l_1 \times m_1} & \cdots & (x_{i_1,l_1})_{l_1 \times m_1} \\
(0)_{m_2 \times 1} & \cdots & (0)_{m_2 \times 1} \\
\vdots & \ddots & \vdots \\
(x_{i_1,l_1})_{l_1 \times m_1} & \cdots & (x_{i_1,l_1})_{l_1 \times m_1} \\
(0)_{m_2 \times 1} & \cdots & (0)_{m_2 \times 1} \\
\end{pmatrix}
\]

\[
\rho^* = \left\{ \begin{array}{ll}
\rho_1(i_1) = a^*, & 1 \leq i_1 \leq l_1, l_1 < l_2 \leq l_2.
\end{array} \right.
\]

**Setup:** \( \mathcal{B} \) chooses \( \omega, \eta \in \mathbb{Z}_p \) and sets \( x = a^* \), \( w = w^* + \alpha \eta^{q+1}, g_1 = g_2 = g, P = (g^\alpha)^\eta, T = e(g, g^\alpha) = e(g, g^{w^*}e(g^\alpha, g^\omega)) \). Then, it chooses \( a_1 = (a_1 + \sum_{\rho^*(t) = a_1} \sum_{t=1}^\infty \alpha^t \omega/m_1) / \eta \).

This implies that:

\[
P_i = g^{a_1} = g^{a^*} \sum_{\rho^*(t) = a_1} \prod_{j=1}^{m_1+m_2-1} (g^{\omega/m_1})^{m_1,m_2}.
\]

Significantly, \( P_i = g^{a^*} \) if and only if \( S_i = \emptyset \).

**Phase 1:** \( \mathcal{B} \) responds to the SK queries. Assume that \( \mathcal{B} \) is given an SK query with a set \( U \), which does not satisfy policy \( (X, \rho_1) \). Then, \( \mathcal{B} \) picks \( \omega' \in \mathbb{Z}_p \) at random. Furthermore, it finds a vector \( \omega = (x_1, \ldots, x_{m_1+m_2}) \in \mathbb{Z}_p^{m_1+m_2} \) such that \( x_1 = 1, \) and \( \forall i, \rho^*(i) \in U, \omega_i M_1 = 0, \) where \( M_1 \) denotes the \( i^{th} \) row of \( M \). By the definition of \( M \), such a vector must exist. \( \mathcal{B} \) sets \( u = \omega + (\sum_{j=1}^{m_1+m_2-1} \alpha^t \omega/m_1) \). Furthermore, it sets:

\[
D_u = g^{a^*} \prod_{j=1}^{m_1+m_2-1} g^{\omega/m_1} \prod_{j=2}^{m_1+m_2-1} g^{-\alpha^2/m_2} \omega / \eta.
\]

Additionally, \( \mathcal{B} \) calculates the attribute SK \( D_i = p_1^{a^*} = g^{a^*} \). We consider the calculation in two cases:

1) For a given \( a_1, \forall \rho^*(t) \neq a_1 \). In this case, \( a_1 = a_1 / x \) and \( D_i = p_1^{a_1} = g^{a_1} / \eta \).

2) For a given \( a_1, \exists \omega' \) such that \( \rho^*(t) = a_1 \). We find \( a_1 = (a_1 + \sum_{\rho^*(t) = a_1} \sum_{t=1}^\infty \alpha^t \omega/m_1) / \eta \).

This implies:

\[
D_i = g^{a^*} \prod_{j=1}^{m_1+m_2-1} (g^{\omega/m_1})^{m_1,m_2} \prod_{k=1}^{m_1+m_2-1} (g^{\omega/m_1})^{m_1,m_2} \prod_{j=1}^{m_1+m_2-1} g^{\omega/m_1} \prod_{k=1}^{m_1+m_2-1} g^{-\alpha^2/m_2} \omega / \eta.
\]

where:

\[
\sum_{\rho^*(t) = a_1} \sum_{j=1}^{m_1+m_2-1} \omega_j x_{1,j}/b_i^* = 0.
\]

Thus:

\[
D_i = g^{a_1} \prod_{j=1}^{m_1+m_2-1} (g^{\omega/m_1})^{m_1,m_2} \prod_{k=1}^{m_1+m_2-1} (g^{\omega/m_1})^{m_1,m_2} \prod_{j=1}^{m_1+m_2-1} (g^{\omega/m_1})^{m_1,m_2} \prod_{k=1}^{m_1+m_2-1} g^{-\alpha^2/m_2} \omega_j x_{1,j}/b_i^*.
\]

Since the unknown term \( g^{a^*} \) is canceled, we can calculate \( D'_u \) and \( D_i \) easily.
Challenge: $B$ builds the challenge ciphertext. $A$ gives two challenge messages $M_0$ and $M_1$ and an appended message $M_2$ to $B$. It creates $C_1 = M_1 \cdot \gamma Y_{1,1}^{\gamma_1}, C_2 = M_2 \cdot T^{\gamma_2}$, and $C_1’ = g^v \cdot C_1$, $C_2’ = g^{v’} \cdot C_2$, where $\gamma_1, \gamma_2 \in Z_p$, and $v \in \{0, 1\}$ are all chosen at random. Then, $B$ chooses $v_2, \ldots, v_m, m_1 + m_2 \in Z_p$ randomly and generates the vector:

$$V_1 = (s + \gamma_1 + y_{1,1}^{y_{1,1}/x_{1,1}^{y_{1,1}}}) + s \sum_{j=1}^{m_1} \frac{y_{1,1}^{y_{1,1} \cdot c_{1+1} + s} \sum_{j=2}^{m_2} \frac{y_{1,1}^{y_{1,1} \cdot v_{m_1+1}}}{x_{1,1}^{y_{1,1}}},}$$

$$\phi + v_2, \ldots, \phi^{m_1 - 1} + v_{m_1}).$$

For each row $X_i$ of $X$, $B$ chooses $r_i’ \in Z_p$ randomly, sets $r_i = r_i’ - sb_i \eta_i$, and calculates:

$$\lambda_i = X_i V_i$$

$$= x_{1,1}^{y_{1,1} \cdot \gamma_1 + y_{1,1}^{y_{1,1}/x_{1,1}^{y_{1,1}}}} + s \sum_{j=1}^{m_1} \alpha^{y_{1,1} \cdot x_{1,1}^{y_{1,1}}} + m_1 y_{1,1} \cdot x_{1,1}^{y_{1,1}} +$$

$$s \sum_{j=2}^{m_2} \alpha^{y_{1,1} \cdot x_{1,1}^{y_{1,1}}} + s \sum_{j=2}^{m_1} v_{m_1+1} + m_1 m_2 \cdot x_{1,1}^{y_{1,1}}, v_{m_1+1} + m_1 m_2 - 1.$$

Assume that $\rho_1(l) = a_i$. We find:

$$\hat{C}_{1,i} = g^{\gamma_{1,i}} = g^{t_i \cdot (g^{sb_1}) - \gamma_{1,i}},$$

$$\hat{C}_{2,i} = g^{\gamma_{1,i} \cdot (g^{sb_1}) - \gamma_{1,i}},$$

Thus, it can easily calculate $\hat{C}_{2,i} = g^{t_i \cdot (g^{sb_1}) - \gamma_{1,i}}$. Furthermore, for $\forall l \neq i_2$, assume that $\rho_2(l) = a_i$. Thus:

$$\hat{C}_{2,i} = g^{y_{1,i} \cdot (g^{sb_1}) - y_{1,i}},$$

Thus, we can get $\hat{C}_{2,i} = g^{t_i \cdot (g^{sb_1}) - \gamma_{1,i}}$. Therefore, $C_{1,i} = g_{1,i}^{y_{1,i} \cdot (g^{sb_1}) - y_{1,i}}$, $C_{2,i} = g_{2,i}^{y_{1,i} \cdot (g^{sb_1}) - y_{1,i}}$, where $c_{1,i}^{y_{1,i} \cdot (g^{sb_1}) - y_{1,i}} \cdot c_{2,i}^{y_{1,i} \cdot (g^{sb_1}) - y_{1,i}}$ are all chosen at random. The public attribute ciphertext of public node $n$ is shown as follows:

$$C_{1,i} = g^{t_i \cdot (g^{sb_1}) - \gamma_{1,i}}, C_{2,i} = g^{t_i \cdot (g^{sb_1}) - \gamma_{1,i}}.$$
B. PERFORMANCE EVALUATION

Scalability and flexibility are two important properties for the IC-IoT data access control mechanism. Different from the traditional access control mechanism, the ABE access control mechanism does not require that a central organization be responsible for managing data access privilege. This implies that ABE has the significant improvement of having access control mechanism scalability. Furthermore, compared with other cryptography schemes, ABE provides a novel one-to-many encrypting pattern, which provides improved flexibility of the IC-IoT data access control mechanism. In brief, the ABE ciphertext access control mechanism is more suitable for the highly pervasive and distributed IC-IoT.

However, the existing ABE scheme has greater ciphertext redundancy, which incurs heavy computation, communication, and storage costs for IC-IoT devices. Thus, low-cost ABE schemes have been extensively researched in recent years. Significantly, in this respect, the proposed CCP-ABE scheme shows comprehensive performance improvement. For performance evaluation, we simulated our CCP-ABE scheme using a Linux virtual machine with 2.83 GHz CPU and 1.00 GB RAM. The result of the simulation is given as follows.

Assume that a set of data \{\textit{M}_1, \textit{M}_2, \ldots, \textit{M}_m\} is given with the access tree set \{\textit{T}_1, \textit{T}_2, \ldots, \textit{T}_m\}. Let \textit{l}_i denote the number of leaves of \textit{T}_i, and let \textit{δ}_i denote the overlap factor of \textit{T}_i, where 1 ≤ \textit{i} ≤ \textit{m}. Furthermore, assume that \textit{l}_i \sim N(\mu_1, \delta_1) and \delta_i \sim N(\mu_2, \sigma_2) are independent and identically distributed. Figure 7 shows the compacted ciphertext size as a function of the number \textit{m} for \textit{μ}_1 = 20, \textit{δ}_1 = \delta_2 = 5, and \textit{μ}_2 = 0, 5, 10, 15. Note that, the curve \textit{μ}_1 = 0 approximates to the uncompacted case, it can be viewed as the reference curve. However, in this case, the ciphertext size increases with the total number of access policies \textit{m}, and its increase rate depends on the expected overlap factor \textit{μ}_2 of each data access policy.

Similarly, assume a ciphertext compacting ratio \( \text{R} = 1 - \text{L}' / \text{L} \), where \text{L}' is the number of leaves in the compacted access tree sets and \text{L} is the number of leaves in the uncompacted access tree sets. Figure 8 shows the compacting ratio of such a ciphertext set as a function of the number \textit{m} when \textit{μ}_1 = 20, \textit{δ}_1 = \delta_2 = 5, and \textit{μ}_2 = 0, 5, 10, 15. Note that, the curve \textit{μ}_2 = 0 approximates to the case of the uncompacted scheme, it can be viewed as the reference curve. In this case, the compacting ratio \text{R} increases quickly with \textit{m} when \textit{m} is small and approaches the constant \textit{μ}_2 / \textit{μ}_1 when \textit{m} is larger than a certain threshold.

Furthermore, Figure 9 shows the encryption time of such compacted ciphertexts as a function of the number \textit{m} when \textit{μ}_1 = 20, \textit{δ}_1 = \delta_2 = 5, and \textit{μ}_2 = 0, 5, 10, 15. Note that, the curve \textit{μ}_2 = 0 approximates to the case of uncompacted ABE scheme, it can be viewed as the reference curve. In this case, the encryption time linearly grows with the number of access policies, but the larger \textit{μ}_2 incurs a lower growth rate.
VI. CONCLUSION

In order to reduce ciphertext redundancy, we provide a policy-compacting method for ABE. The method can reduce various overheads of the ABE scheme without sacrificing any additional performance. However, the policy-compacting problem is an NPC problem, and a greedy compacting algorithm is provided to achieve an approximate minimum ciphertext scale. The detailed security proof and performance evaluation of the scheme are also given in this work. These demonstrate that the scheme obtains comprehensive performance improvement, with its storage and computation overhead all significantly reduced.

APPENDIX A FLEXIBILITY FACTOR

Let n be a non-leaf (t_n-out-of-n node) of access tree T, p be the parent of n, γ_n be the flexibility factor of n, γ_n denote the upper limit of γ_n, and p be a prime. Here we prove that the inequality in Equation (13) holds. First, γ_n is discussed in two cases:

1) If t_n ≠ 2. This case is presented in two further subcases: t_n = 1 and t_n = 3. If t_n = 1, f_n(x) must be a constant function, and it can be affirmed with only one interpolating node (x_c, s_c), c ∈ N_n. Thus, γ_n = t_n = 1. If t_n ≥ 3, let f_n(x) be a (t_n-1)-order polynomial function. It is clear that it is highly possible that there is a non-feasible solution x ∈ Z_p for f(x) ≡ s (mod p), s ∈ Z_p when t_n ≥ 3. For feasibility and efficiency, f(x) is constructed as a Lagrange interpolation polynomial:

\[ f(x) = \sum_{i \in N_n} \Delta_{s_i, x, N_n}(x) s_i, \]

where s_i ∈ Z_p is chosen at random, N_n ⊆ N and |N| = t, X = \{x_c, c ∈ N\}, and \( \Delta_{s_i, x, N_n} \) is the Lagrange coefficient of c. Note that the shares of c ∈ N are not all equal, otherwise f(x) is degenerate and hence a constant function. For the remaining nodes c_i ∈ N_n - N, the node share s_i is set to f(x_c), where x_c is chosen at random. Thus, γ_n is also equal to t_n in this subcase.

2) For t_n = 2. Let s_c ∈ Z_p be the share of c ∈ N, and the polynomial can be assigned as f_c(x) = ax + s_c, where a ∈ Z_p and s_c is the share of n. However, all shares of s_c, 0 ≤ i ≤ n do not equally guarantee that f_c(x) is not degenerate. ∀c_i, 1 ≤ i ≤ n, the valid interpolation in Z_p can be calculated as \( x_i = a^{-1}(s_c - s_n) \mod p \). Thus, γ_n is generally set to be |N_n| when t_n = 2.

Next, the inequality \( \sum_{n \in T} \gamma_n \leq |\bigcup_{n \in T} N_n| \) is considered. Let n be a non-leaf of T, h_n be the height of n, and N_n = \{c ∈ N_n | γ_c ≥ |N_n|\}. The equation \( \gamma_n = \min\{\gamma_c, |N_n|\} \) can be inductively proved. If h_n = 2, then N_n = N_n'. Thus, γ_n = \min\{γ_c, |N_n'|\} holds.

Assume that \( \gamma_n = \min\{\gamma_c, |N_n'|\} \) when h_n ≤ k. Then, we prove that \( \gamma_n = \min\{\gamma_c, |N_n'|\} \) when h_n = k + 1. ∀T' ⊆ T, \( \sum_{n' \in T'} \gamma_{n'} \leq |\bigcup_{n' \in T'} N_{n'}| \). This implies that:

\[ \gamma_n = \min\{|U_{n' \in T_n'} N_{n'}| = \sum_{n' \in T_n'} \gamma_{n'}, |N_{n'}|\}. \]

where |T_n'| denotes that a subtree of T includes n. Thus, we select a subtree T_n ⊆ T that satisfies the following conditions:

1) n is the root of T_n; and
2) ∀c ∈ N_n', n' ∈ T_n, if γ_c ≥ |N_n'|, then c ∈ T_n.

∀n' ∈ T_n - {n}, we have h_{n'} ≤ k and γ_{n'} = \min\{γ_{n'}, |N_{n'}|\} ≥ |N_{n'}|.

As a result:

\[ \gamma_n = \min\{|U_{n' \in T_n'} N_{n'}| = \sum_{n' \in T_n'} \gamma_{n'}|N_{n'}|\}. \]

APPENDIX B OVERLAP FACTOR

Let T be an access tree, N_T be the non-leaf set of T, τ be the root of T, and d_T describe the depth of T. Here, we prove that Equation (14) holds. Considering security, the root τ of T must be assigned with a random secret. Thus, the overlap factor δ_T is analyzed as follows.

If k = 2, then τ is the only non-leaf of T. We describe δ_T in two cases:

1) s_i is assigned. Thus, δ_T = γ_n - 1. As a result:

\[ \delta_T = \sum_{n' \in N_T} (\gamma_{n'} - 1). \]

2) s_i is unassigned. Then δ_T is calculated as follows:

\[ \delta_T = \left\{ \begin{array}{ll} \gamma_n, & \gamma_n < N_n' \\ \gamma_n - 1, & \gamma_n = N_n' \end{array} \right. \]

This implies that:

\[ \delta_T = \left\{ \begin{array}{ll} 1 + \sum_{n' \in N_T} (\gamma_{n'} - 1), & \gamma_n < N_n' \\ \sum_{n' \in N_T} (\gamma_{n'} - 1), & \gamma_n = N_n' \end{array} \right. \]

Assume Equation (14) always holds when δ_T ≤ k. Then, we prove that the equation hold when δ_T = k + 1. Let T_{c_i} be the i'th subtree of τ, where c_i ∈ N_T - {τ} and d_{τ,c_i} ≤ k. δ_T is also discussed in two cases:

1) s_i is assigned. For an arbitrary subset N̄_T ⊆ N_T - {τ}, |N̄_T| = γ_τ - 1, ∀c_i ∈ N̄. Let N̄_T be assigned with random node share (i.e., s_i is unassigned). At the same time, ∀c_i ∈ N̄_T - {τ}, s_i must be assigned. Thus:

\[ \delta_{τ,c_i} = \left\{ \begin{array}{ll} 1 + \sum_{n' \in N_T} (\gamma_{n'} - 1), & c_i \in N̄_T \\ \sum_{n' \in N_T} (\gamma_{n'} - 1), & c_i \in N̄_T - {τ} \end{array} \right. \]

As a result, we calculate:

\[ \delta_T = \sum_{c_i \in N_T} \delta_{τ,c_i} + \sum_{c_i \in N_T - {τ}} \delta_{τ,c_i} + \sum_{c_i \in N_T - N_T} \delta_{τ,c_i} \]

\[ = N̄_T + \sum_{c_i \in N_T - {τ}} \delta_{τ,c_i} \]

\[ = 1 + \sum_{n' \in N_T} (\gamma_{n'} - 1). \]

2) s_i is unassigned. For an arbitrary subset N̄_T ⊆ N_T - {τ}, |N̄_T| = 0, ∀s_i, c_i ∈ N̄_T is unassigned and ∀c_i ∈ N̄_T - N̄ - {τ}, s_i is assigned. Then, δ_T is also calculated as the equation shows. Thus, we find:

\[ \delta_T = N̄_T + \sum_{c_i \in N_T - {τ}} \delta_{τ,c_i} \]

\[ = \min\{|γ_c, N̄_T - {τ}|\} + \sum_{n' \in N_T - {τ}} (\gamma_{n'} - 1). \]

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Citation information: DOI 10.1109/ACCESS.2018.2877177, IEEE Access.
and:

$$\delta_T = \left\{ \frac{1 + \sum_{\omega \in \omega^N (\gamma_\omega - 1)} \cdot \gamma_T < \gamma_{\omega^N} \cdot \sum_{\omega \in \omega^N (\gamma_\omega - 1)} \cdot \gamma_N = \gamma_{\omega^N} \right\}$$

REFERENCES


JING WANG is a PhD of Wuhan University and a post-doctor of Sun Yat-sen University. Her research interests include security and privacy of cloud.

NEAL N. XIONG is current an associate professor at Department of Mathematics and Computer Science, Northeastern State University. He received his both PhD degrees in Wuhan University (about software engineering), and Japan Advanced Institute of Science and Technology (about dependable networks), respectively. Before he attends Northeastern State University, he worked in Wentworth Technology Institution, Georgia State University for many years. His research interests include Cloud Computing, Security and Dependability, Parallel and Distributed Computing, Networks, and Optimization Theory.

WANG JINHAI is a PhD of Wuhan University and an associate professor of Foshan University. His research interests include cloud computing and high performance computing.

WEI-CHANG YEH received his M.S. and Ph.D. from the Department of Industrial Engineering at the University of Texas at Arlington. He is currently a distinguished professor in the Department of Industrial Engineering and Engineering Management at National Tsing Hua University in Taiwan. His research interests include network reliability theory, graph theory, the deadlock problem, linear programming, and scheduling. He is a member of INFORMS, and a senior member of the IEEE.