CHAPTER

5 Lightweight ciphertext-policy attribute-based encryption scheme for data privacy and security in cloud-assisted IoT

Syam Kumar Pasupuleti*, Dheerendra Varma†
**IDRBT, Hyderabad, India †University of Hyderabad, Hyderabad, India*

5.1 Introduction

The Internet of things (IoT) is a network of physical devices that have unique IP addresses and the ability to sense and collect data from the external world. In IoT the devices exist in different geographic locations, and information collected through the devices can be used for different applications in areas such as healthcare, banking, and others [1]. Due to lack of memory, processing power, and bandwidth, IoT devices can't handle large and complex data. Therefore the cloud is the best option for storing this type of data as it provides scalable storage capacity under a pay-per-use model.

The cloud provides resources such as large storage space and processing capabilities at low costs to enable the use of real computing machines. The cloud is also useful for hosting websites and mobile applications for increasing resources to achieve scalability. However, moving data to the cloud brings two main concerns: (1) privacy and (2) access control [2]. For this reason, we need to ensure data privacy and provide access control for authorized users of cloud-assisted IoT.

The traditional solution to protect the privacy of data is encryption, but encryption schemes do not provide access control. To provide access control, various traditional approaches have been proposed, including mandatory access control (MAC), discretionary access control (DAC), and role-based access control (RBAC). In MAC, security labels are assigned to all the resources and users in the organization. Whenever a user wants to access the resources the operating system checks the user security label with the resource security label. If the two match, then resource access is allowed. If not, access is denied. The problem with this model is the amount of preplanning necessary before it can be effectively implemented. Management overhead for this model is high as security labels need frequent updating. In DAC, which is based on an access control list (ACL), users control their own data. A user gives the permission to access the system resources. The user's access

Real-Time Data Analytics for Large Scale Sensor Data. https://doi.org/10.1016/B978-0-12-818014-3.00005-X

permission details are stored in the ACL. DAC is not suitable for use when the data is stored in distributed environments. In RBAC, all users are assigned roles based on their job function in the organization. Resource access is based on these assigned roles. If we want to suspend a user, we have to suspend the entire group of users assigned the same roles. Therefore RBAC is not optimal because large groups would have the same type of access. Although all these methods provide access control, they do not provide data privacy.

Attribute-based encryption (ABE) overcomes the issues of traditional access control methods and provides data privacy as well [3]. There are two variations of ABE: key-policy attribute-based encryption (KP-ABE) and ciphertext-policy attribute-based encryption (CP-ABE). CP-ABE is the best approach for ensuring data privacy and fine-grained access control in the cloud environment because the data owner has control over the access policy.

In CP-ABE, data owners encrypt the data under the access policy. The data can be decrypted by users only if their attributes fulfill the owner-defined access policy. Recently several CP-ABE schemes were proposed based on different access structures. These include access trees [4–9], AND gates [10–13], and linear secret sharing schemes (LSSS) [3, 14–16]. However, all these existing CP-ABE schemes increase computation overhead for data encryption due to their access structure representation.

To make CP-ABE more efficient, we need to focus on the access structure [17]. These drawbacks motivated us to design a lightweight ciphertext-policy attribute based encrption (LCP-ABE) scheme with a new access structure. This chapter presents and discusses our LCP-ABE scheme for ensuring data privacy using cloud-assisted IoT.

In our LCP-ABE, we use a Walsh–Hadamard transform for data privacy and access. Our scheme supports efficient user revocation and is secure against collusion attacks, chosen-plaintext attacks (CPA), and forward secrecy (FS). It is also lightweight and more suitable for IoT devices.

In this chapter we describe related work (Section 5.2), discuss preliminaries (Section 5.3), define our system model (Section 5.4), present construction of the LCP-ABE scheme (Section 5.5), analyze the system's security (Section 5.6), and conclude our findings (Section 5.7).

5.2 Related work

Many researchers have proposed CP-ABE approaches based on different access structures to ensure data privacy and access control. As discussed in the previous section, there are three types of CP-ABE schemes: (1) access trees with threshold gates, (2) AND gates, and (3) linear secret sharing schemes (LSSS).

5.2.1 CP-ABE scheme with threshold gates

Bethencourt et al. [5] were first to design a CP-ABE scheme based on an access tree with threshold gates and encrypted message stores. In their scheme, decryption of the information can be done using Shamir's Secret Sharing [18]. Goyal et al. [6]

proposed a "bounded CP-ABE" that generalizes transformation from KP-ABE to CP-ABE through a universal access tree. This scheme accepts any access policies with bounded polynomial size but this scheme has a drawback that user is strictly use the depth of the access tree. Ibraimi et al. [9] designed a new method by using n-array access tree with AND and OR nodes, but the decryption operation does not work based on secret sharing. Guan et al. [7] designed a scheme in which the encryption algorithm divides the data into chunks and encrypts it in sequence with its corresponding subtree. Alrawais et al. [4] used the CP-ABE scheme for designing key exchanging and establishing a secure communication among the fog nodes. Hur and Noh [8] addressed the issues of attribute and user revocation. In their scheme, the service provider generates the key encryption keys (KEK) for each user after the setup phase is complete. The generated ciphertext is re-encrypted by using the attribute group key and this key is also encrypted with KEK. Hence unauthorized users are unable to perform the decryption operation. However, it is difficult to maintain all KEK for a large number of users in an organization.

Although threshold-based schemes achieve data privacy and access control, they are not suitable for lightweight IoT devices.

5.2.2 CP-ABE scheme with AND gates

AND gates follow the bit-wise representation of attribute presence in the access policy. Several researchers have proposed CP-ABE schemes based on AND gates. Cheung and Newport [10] described a new CP-ABE by using the AND gate access structure on "positive and negative attributes." Nishide et al. [12] proposed a CP-ABE scheme to hide the "access policy" with partially hidden access structures. Similarly, Emura et al. [11] proposed a scheme that enables a constant ciphertext length with AND gates based on multi-value attributes. Guo et al. [19] discussed a CP-ABE scheme based on AND gates that maintains constant-size decryption keys and is suitable for lightweight devices. Yu et al. [13] first used a scheme with a combination of both CP-ABE and proxy re-encryption (PRE). In this scheme, the authority has re-encryption rights and the ability to update the secret key.

These schemes generate constant-size decryption keys due to their AND-gate structure, but the structure is not expressive enough to hold all conditions.

5.2.3 CP-ABE scheme with LSSS

LSSS is represented as a form of matrix for the attributes in the system. The following CP-ABE schemes were proposed using LSSS gate access structure.

Waters [3] proposed a new methodology of expressing access control with the help of LSSS matrix over the attributes. In this scheme, both encryption and decryption costs are linearly increased because of the access structure's high complexity. It is similar in performance to the scheme proposed by Bethencourt et al. [5]. The algorithm proposed by Waters [3] unnecessarily creates more rows in a matrix from the Boolean formula. Liu and Cao [15] proposed a new procedure to decrease the total number of rows in the LSSS matrix by including Boolean formula as a labeled function to ciphertext. This drastically reduces the costs of communication. Another new scheme designed by Hu et al. [14] improves the efficiency of decentralized ABE scheme with the help of a smaller LSSS matrix. In this matrix, they used better pairing and multi-pairing operations. Naruse et al. [16] described another CP-ABE scheme to address attribute revocation in LSSS access structure using re-encryption done by the cloud. In this scheme, they generate the secret key components for granting attributes instead of generating a new secret key for all attributes.

All these LSSS access structure schemes are in the form of matrices, which increases the size of the ciphertext.

All of the schemes discussed [3–16, 20, 21] are lacking in efficiency due to their access structure representation and are not suitable for IoT devices. Table 5.1 presents a detailed comparison of different ABE schemes.

Table 5.1 Different ABE schemes with different access structures

Scheme	Access structure	Security	Privacy	Access control	Revocation
Bethencourt et al. [5]	Access tree	Selective security	✓	✓	×
Goyal et al. [6]	Bounded access tree	Selective security	✓	✓	×
Ibraimi et al. [9]	Access tree	Selective security	✓	✓	×
Alrawais et al. [1]	Access tree	Selective security	✓	✓	×
Hur and Noh [8]	Access tree	Selective security	✓	✓	✓
Cheung and Newport [10]	AND gates	Selective security	✓	✓	×
Guo et al. [19]	AND gates	Selective security	✓	✓	×
Nishide et al. [12]	AND gates	Selective security	✓	✓	×
Emura et al. [11]	AND gates	Selective security	✓	✓	×
Lewko et al. [20]	LSSS	Full security	✓	✓	×
Waters[3]	LSSS	Selective security	✓	✓	×
Lewko and Waters [21]	LSSS	Selective security	✓	✓	×
Naruse et al. [16]	LSSS	Selective security	✓	✓	✓
Ours(LCP-ABE)	Walsh–Hadamard Transform	Selective security	✓	✓	✓

5.3 Preliminaries

5.3.1 Bilinear maps

Let G_1 and G_T be two multiplicative cyclic groups of prime order p. Let g be a generator of G_1 and e is bi-linear map $e: G_1 \times G_1 \rightarrow G_T$, which has the following properties:

Bilinearity: for all $g, g \in G_1$ and $a, b \in Z_p$, $e(g^{a}, g^{b}) = e(g, g)^{ab}$.
Non-degeneracy: $e(g, g) \neq 1$.

5.3.2 CP-ABE framework

Setup: This setup algorithm can take the universal attributes as one parameter, and executed by the key authority it will generate public parameters and a master key.
Encrypt: This encryption algorithm takes the message M and encrypts it by using public parameters and sends it to the cloud via the access structure.
KeyGen: This KeyGen algorithm is executed by the key authority and generates the user's secret key from the user's attribute set.
Decrypt: This algorithm is executed by the user by using the user's secret key from the key authority and the ciphertext received from the cloud.

5.3.3 Access structure

Definition: Let $\{P_1, P_2, \ldots P_n\}$ be the set of parties. A collection $A \subseteq 2^{\{P_1, P_2, \ldots P_n\}}$ $A \subseteq 2^{\{P_1, P_2, \ldots P_n\}}$ is monotone if $\forall Q, R$: if $Q \in A$ and $Q \subseteq C$ then $C \in A$. A monotic access structure is a collection of A of non-empty subsets of $\{P_1, P_2, \ldots P_n\}$, that is $A \subseteq 2^{\{P_1, \ldots P\}} \backslash \{\varphi\}$. The sets in A are called authorized sets; otherwise unauthorized sets.

In our scheme, we use the Fast Walsh–Hadamard Transform (FWHT) [22, 24] to compute the access structure from the access policy. The FWHT is a divide-and-conquer algorithm that recursively breaks down a truth table of size n into a smaller size, $n/2$. This algorithm takes only $O(n \log n)$ additions or subtractions. We present the construction of the access structure in Section 5.4.3.

5.4 System model

5.4.1 System architecture

Fig. 5.1 represents the system architecture of our scheme. We assume that the communication links from the trusted authority are secure. In our scheme, the following blocks are useful: IoT devices, cloud, -trusted authority, and users. IoT devices are physical objects with internet connectivity that can communicate with other devices over the Internet. These devices perform encryption operations to obtain the ciphertext and transfer it to the cloud for storage purposes. The cloud stores the ciphertext and gives access to users. A trusted authority (TA) generates the keys for encryption and decryption. Then the user gets the ciphertext from the cloud and decrypts it locally.

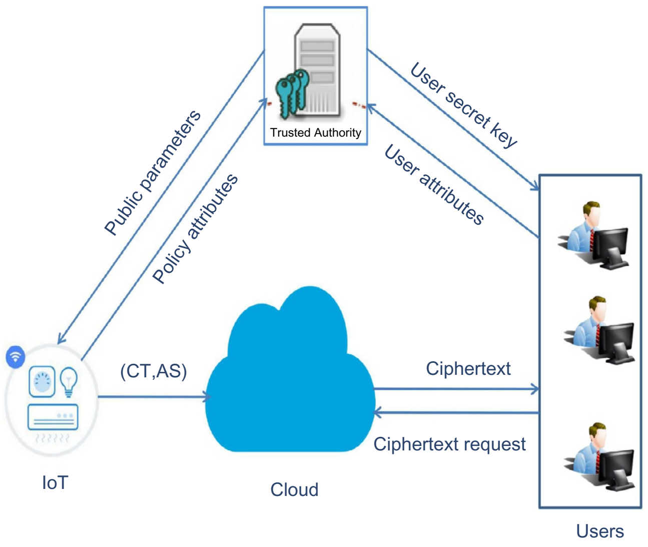


FIG. 5.1

System architecture.

5.4.2 Design goals

We designed a lightweight CP-ABE scheme to achieve the following goals:

- *Privacy*: The data should not disclose to unauthorized persons from the cloud.
- *Access control*: The users can access data if their attributes satisfy the access policy.
- *User revocation*: If any user is removed from the organization, that revoked user should not be able to get the information from the cloud.
- *Lightweight*: The LCP-ABE scheme takes less time for encryption and decryption operations.

5.4.3 LCP-ABE algorithm definitions

The LCP-ABE scheme contains seven algorithms:

- *Setup* ($\mathcal{N} \dashrightarrow$ PK, MSK): It takes input $\mathcal{N}$ as a list of universal attributes and returns a public key PK and master secret key MSK.

- *KeyGen* (MK, $A \dashrightarrow$ USK): It takes the input as the current MSK, user Attributes and outputs the User secret key USK.
- *Encrypt* (PK, M, WHT $\dashrightarrow$ CT): It takes input as the PK, message M, access policy represented in the form of access structure WHT and the generates the ciphertext as the CT.
- *ReKeyGen* (δ, MSK $\dashrightarrow$ rk, PK$'$, MSK$'$): It takes attribute list δ contains the list of the attributes used for update andMSK as input. It returns the new public key PK$'$, new master secret key MSK$'$, and rekey rk.
- *ReEncrypt* (CT, rk, $\delta \dashrightarrow$ CT$'$): It takes input as the CT, revocable attribute list δ, and the revocable keys rk. It outputs the new ciphertext CT$'$ according to the attributes in δ and by using the revocable keys rk.
- *KeyUpdate* (USK, rk, $\delta \dashrightarrow$ USK$'$): It takes the input as USK and revocable attribute list δ and outputs the new modified user secret key USK$'$.
- *Decrypt* (CT, PK, USK $\dashrightarrow M$): It outputs plain text message M based on CT, PK, and the USK as input.

Fig. 5.2 shows the flow of the LCP-ABE algorithms.

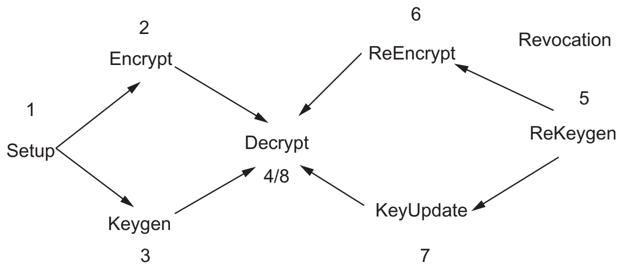


FIG. 5.2

Flowchart of LCP-ABE.

The order of execution of flowchart LCP-ABE algorithms is as follows:

1. TA runs a setup to generate PK and MSK, then sends PK to IoT devices
2. IoT device runs an encrypt algorithm and sends ciphertext along access policy to cloud for storage
3. TA produces a private key for user attributes by running KeyGen algorithm
4. The user runs a decrypt algorithm to decrypt the data based on private key
5. Whenever a user is revoked from system, TA runs ReKeyGen algorithm to generate rekey for re-encryption of ciphertext
6. Proxy server re-encrypts the ciphertext using rekey and sends to cloud
7. Simultaneously, the non-revoked user updates their private keys
8. The user runs a decrypt algorithm to get the plain text using updated secret key

5.4.4 Security game

To prove security strength of LCP-ABE, we define a security game between the Adversary ***Adv*** and a challenger ***Sim*** as follows:

- *Initialization*: The ***Adv*** randomly selects an access structure WHT and transfers it to the ***Sim***.
- Setup: The ***Sim*** executes **setup** phase to generate the public parameters PK and sends to the ***Adv***.
- *Phase 1*: The ***Adv*** sends a query to the **KeyGen** phase by using the user attribute set **A** to get the user secret key USK. But the restriction to this user attribute set is **A** $\nVdash_{\text{WHT}}$ must hold. Then ***Sim*** sends the USK to ***Adv***.
- *Challenge*: The ***Adv*** sends same length of messages M_0 and M_1 to the ***Sim***. Then ***Sim*** randomly selects $\gamma \in \{0, 1\}$ and encrypts M_γ using $_{\text{WHT}}$ to get the ciphertext CT. Finally, *Sim* sends CT along with access structure WHT to the ***Adv***.
- *Phase2*: ***Adv*** repeats actions similar to *Phase 1*.
- *Guess*: The ***Adv*** guesses the value of γ as γ'. The advantage of ***Adv*** is:

$$Adv^{\text{CPA}}_{\text{CP-ABE}}(A) = \left|\Pr[\gamma = \gamma'] - \tfrac{1}{2}\right|$$

5.5 Construction of LCP-ABE

In this section, we construct the proposed LCAP-ABE scheme based on defined algorithms in Section 5.4.3 as follows:

5.5.1 Setup ($\mathbb{N} \dashrightarrow$ PK, MSK)

In this algorithm, TA generates PK and MSK as follows:

i. Select a generator g of group G_1 with bilinear map $e: G_1 \times G_2 \rightarrow G_T$
ii. Randomly select $y, t_1, t_2, \ldots t_n \in Z_p$
iii. Compute $Y = e(g, g)^y$, $T_i = g^{t_i}(i \in N)$ for every attribute $i \in N$
iv. Public key PK $= (e, g, Y, T_i)$ and Master secret keyMSK $= (y, t_i)$

5.5.2 KeyGen (MK, $A \dashrightarrow$ USK)

This algorithm perform the following steps to generate the USK from the attribute set A $= \{a_1, \ldots a_n\}$

i. Generate a random number $r \in \mathbb{Z}_p$
ii. Generate bit string (B) from the following formula:

```
for i∈ I
        if i∈ A ∧ i = i then $b_i = 1$
            else $b_i = 0$ then
    Bit String B = {$b_1$, $b_1$ ,… $b_n$}
```

For example attributes derived from the policy are ['hcu','idrbt','mtech'] and user attributes are ['hcu','mtech'] then the bit string(B) after computation will be [1,0,1].

iii. Compute $D_1 = g^{y-r}$, $D_w = g^{rt_i}$ where $i \in I$

iv. The USK $= (B, D_1, D_w)$

5.5.3 Encryption (PK, M, WHT ⇢ CT)

This algorithm executes in two phases. The first phase constructs the access structure for the given access policy and the second phase generates the ciphertext.

5.5.3.1 Access structure (policy→ WHT)

This algorithm takes input as access policy and converts it into the WHT. The following steps are used in the construction of the access structure.

i. Convert the access policy into the Boolean function.

for example: ((hcu and idrbt) or (hcu and mtech) or (mtech and idrbt)) = ((x1 $\wedge$ x2) $\vee$ (x1 $\wedge$ x3) $\vee$ (x3 $\wedge$ x2)).

ii. Extract the attributes I from the access policy I $\subset$ N and construct the truth table for the given Boolean function.

for example: From this Boolean function f(x) = ((x1 $\wedge$ x2) $\vee$ (x1 $\wedge$ x3) $\vee$ (x3 $\wedge$ x2)) converts to the truth table (TT).

x1	0	0	0	0	1	1	1	1
x2	0	0	1	1	0	0	1	1
x3	0	1	0	1	0	1	0	1
f(x)	0	0	0	1	0	1	1	1

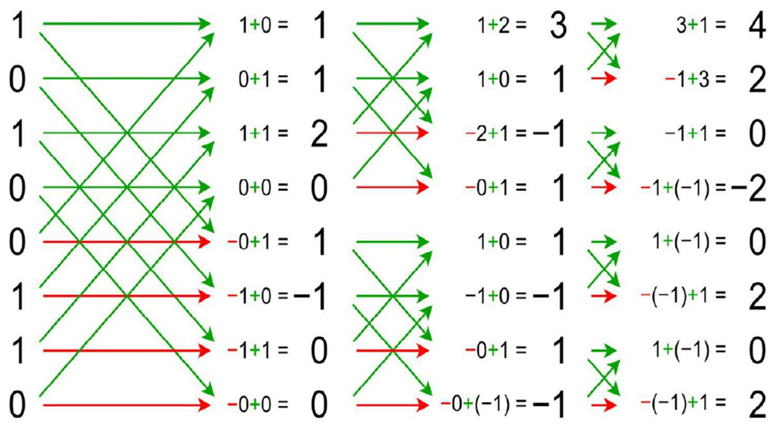
This table contains the all possible cases of attributes in access policy.

iii. From the TT, we construct the access structure as described in Algorithm 1.

Algorithm 1 Fast Walsh-Hadamard transform

```
1:  Procedure: FWHT(TT )
2: n ←TT:SIZE()
3: WHT[n]=TT
4:   for i : 1 → (n/2-1) do
5:       for j : 0 → n  i : do
6:           WHT[j] := W HT [j] + WHT [j + i]
7:           WHT [j] := WHT [j] - W HT [j + i]
8: return (W HT )
```

iv. The pictorial representation of this conversion is shown in the below figure.



5.5.3.2 *Encrypt (PK, M, WHT → CT)*

After construction of the access structure, the encryption algorithm performs the following steps to generate the ciphertext:

i. Randomly select $s \in Z_p$.
ii. Compute $C_1 = M.\ Y^s$, $C_2 = g^s$, $C_w = 1/\left(\prod_{i\in I} T_i\right)$ and the ciphertext element $C_3 = (C_W)^s$.
iii. The ciphertext CT = (WHT, C_1, C_2, C_w, C_3) send to cloud.

5.5.4 ReKeyGen(δ, MSK ⇢ rk)

This ReKeyGen algorithm generate the *rk* as follows:

i. For each attribute in $i \in \lambda$ randomly choose $t'_1 \in Z_p$
ii. Compute $\text{rk}_i = {}^{t'_i}/_{t_i}$ otherwise $\text{rk}_i = 1$
iii. Return the proxy re-key $\text{rk} = \text{rk}_i (i \in N)$ and also
iv. Return new PK' and MSK'

5.5.5 ReEncrypt (CT, rk, δ ⇢ CT′)

This algorithm modifies the ciphertext related to the revoked user to generate new ciphertext CT' as follows:

i. Compute the new attribute related ciphertext component $C'_w = \left(\prod_{i\in I} T_i^{rt_i}\right)^s$ and $C'_3 = (C'_w)^s$
ii. Output the new ciphertext $\text{CT}' =$ (WHT, C_1, C_2, C'_w, C'_3)

5.5.6 KeyUpdate (USK, rk, $\delta \dashrightarrow$ USK′)

The non-revoked users to update their User secret key USK’ as follows:

```
for each $i \in I$
  $D_w = D_w + (t_i \times \mathrm{rk}_i)$
Then $D_2 = g^{(r/D_w)}$
```

The remaining secret key components are kept the same and output the new USK

USK’ $= (B, D_1, D'_2)$

5.5.7 Decrypt (CT, PK, USK $\dashrightarrow$ M)

In this algorithm, the user decrypts the ciphertext using the secret key after getting ciphertext from the cloud as follows:

i. Compute row of WHT by using the following formula:

```
$H_m[B]$:$B^{th}$ row of the Hadamard matrix
  for i := 0 to WHT. size() - 1
   $H_m[B] = (B,i)$
```

where $B = (p_1, p_2, p_3, \ldots, p_n)$ and $i = (q_1, q_2, q_3, \ldots, q_n)$ are n dimensional binary vectors and $(B,i) = p_1 q_1 + p_2 q_2 + \ldots + p_n q_n$ are i the inner product of vectors B and i.

ii. Compute the acceptance of user attributes by using value of ACC:

$$\mathrm{ACC} := \sum_{j=0}^{\mathrm{WHT.size()}} (\mathrm{WHT}[j] \times H_m[B][j] / \mathrm{WHT.size}()$$

iii. if ACC $= 1$ then it means that user satisfies the access policy and decryption of the message M as follows:

$$Y_s = e(C_2, D_1).e\left(C_3, \prod_{i \in I} D_i\right) Y_s := e(C_2, D_1).e(C_3, \prod_{i \in I} D_i)$$
$$M = C_1 / Y_s)$$

iv. Otherwise, decryption of the message M is not possible because the User attributes do not satisfy the access policy.

5.6 Security analysis

In this section, we prove the correctness, security strength, and forward secrecy of the LCP-ABE.

5.6.1 Correctness

The correctness of the decryption algorithm is proved as:

$$\mathrm{M} = \frac{C_1}{\ell(C_2, D_1) \times \ell(C_w, D_2)}.$$

$$= \frac{M.Y^s}{\ell(g^s, g^{y-r}) \times \ell\left(\left(\prod_{i\in I} T_i\right)^s, g^{\frac{r}{\sum_{i\in I} t_i}} \right)}$$

$$= \frac{M.\ell(g,g)^{ys}}{\ell(g,g)^{ys-rs} \times \ell\left(g^{\sum_{i\in I} t_i . s}, g^{\frac{r}{\sum_{i\in I} t_i}} \right)}$$

$$= \frac{M.\ell(g,g)^{ys}}{\ell(g,g)^{ys-rs} \times \ell(g,g)^{\sum_{i\in I} t_i . s \cdot \frac{rs}{\sum_{i\in I} t_i}}}$$

$$= \frac{M.\ell(g,g)^{ys}}{\ell(g,g)^{ys-rs} \times \ell(g,g)^{rs}}$$

$$= \frac{M.\ell(g,g)^{ys}}{\ell(g,g)^{ys-rs+rs}}$$

$$= \frac{M.\ell(g,g)^{ys}}{\ell(g,g)^{ys}}$$

$$= \mathrm{M}$$

5.6.2 Security strength

Here, we analyze the security strength of our scheme against two attacks: collusion attack and chosen plain text attack.

5.6.2.1 Collusion attack

In our LCP-ABE scheme, we construct the access structure for each encrypted data and stores in the cloud. The decryption process requires the subset of attributes described by the user. USK generation involves the random number r in the key-generation algorithm for every user. Therefore illegal users can't get the data because the random number is different for every user.

5.6.2.2 CPA security proof

Here, we prove the security strength against CPA through DBDH assumption

Theorem 1 *Suppose our scheme holds the DBDH assumption that no adversary can break the LCP-ABE.*

Proof: This theorem can be proved based on the security game defined in Section 5.4.4. as follows:

Initialization: ***Adv*** chooses the WHT in 2^n form and transfers to the ***Sim***.

Setup: ***Sim*** executes setup phase to provide the PK to the ***Adv.*** It takes the input as universal attributes $\aleph = \{A_1, A_2, A_3, \ldots\ldots A_n\}$. ***Sim*** randomly picks the values for each attribute $t_1, t_2, t_3, \ldots t_n$ and generates the PK as $g^{t_1}, g^{t_2}, g^{t_3}, \ldots, g^{t_n}$ $g^{t_1}$and outputs PK and MK as $<\ell, g, T, Y>$ and $<y, t>$.

Phase 1: ***Adv*** sends the user attributes A in USK-generation phase where A $\Vdash\!\!\!\!/$ WHT. So, S must be chosen that A can't satisfy the valid combination in WHT. There should be an attribute $i \in I$ that satisfies either $i \in A$ or $i \notin A$. Then, ***Sim*** chooses attribute that satisfies $i \in I$ and $i \notin S$.

Sim chooses the random value $r \in Z_p$.

Bit string (B) is obtained by following formula:

```
For each attribute $i \in I$:
  $if\ i \in A\ i \wedge i = I\ then\ b_i := 1\ else\ b_i := 0.$
```

Then $B = [b_1, b_2, b_3, \ldots b_n]$.

Then compute secret key values as $D_{1:=}g^{y-r}$, $D_{2:=}g^{\overline{\sum_{i\in I} t_i}^{\,r}}$ and send to ***Adv***.

Challenge: ***Adv*** sends messages with same length M_0 and M_1 $\mathrm{M}_0$and$\mathrm{M}_1$to ***Sim***. ***Sim*** randomly picks one element from $\mu \in \{0, 1\}$ $\{0,1\}$ and selects the $C_1 = \mathrm{M}_\mu$. Z and generates the ciphertext CT*:= (WHT, C_1,C_2,C_w) CT^* WHT, C_1, C_2, C_w send to ***Adv***.

Phase 2: ***Adv*** repeats *Phase 1*.

Guess: ***Adv*** guess a output γ' of γIf $\gamma' = \gamma$ ***Sim*** generates the output 0 to indicate that $Z = g(g,\ g)^{\mathrm{abc}} g^{\mathrm{abc}}$ is a valid ciphertext, so the advantage of Adv is ε. Thus

$$\mathrm{P_r}[\mathrm{Sim} \rightarrow \text{“0”} \mid \mathrm{Z} = e(g,g)^{\mathrm{abc}}] = \mathrm{P_r}[\gamma' = \gamma \mid \mathrm{Z} = e(g,g)^{\mathrm{abc}}] = \tfrac{1}{2} + \varepsilon$$

If $\gamma' \neq \gamma$, then sim outputs 1 to indicate $\mathrm{Z} = e(g,g)^{\mathrm{z}}$. Then it's completely random from ***Adv***. Thus.

$\mathrm{P_r}[\mathrm{Sim} \rightarrow \text{“1”} \mid Z = e(g,g)^{z}] = \mathrm{P_r}[\gamma' = \gamma \mid Z = e(g,g)^{z}] = \tfrac{1}{2}$.

Based on the preceiding analysis, the advantage of ***Adv*** to break our scheme is:

$$\tfrac{1}{2}\left(\tfrac{1}{2} + \varepsilon + \tfrac{1}{2}\right) - \tfrac{1}{2} = \tfrac{\varepsilon}{2}.$$

Hence, any polynomial time adversary cannot break the LCP-ABE scheme.

5.6.3 Forward Secrecy

Our scheme guarantee the forward secrecy of the IoT data against any users contraining the revoked users. Forward secrecy means that session keys will not be compromised even if the private key of the server is compromised. Revoked users cannot access the data in the future after the revocation happens. In our scheme, after the revocation of the particular user, the trusted authority recomputes the master key, public parameters, and RK. The proxy updates ciphertext components and generates the new ciphertext. Thus, revoked user who can have been revoked, cannot able to find the blinding factor.

$$e(C_2, D_1) \times e\ (C_w, D_2)$$

So, the forward secrecy of the IoT data against revocation is assured.

5.7 Performance analysis

This section measures the communication and computation overhead theoretically.

5.7.1 Communication overhead

Here, we mainly focus on size of ciphertext. The secret keys are generated after the execution of encryption and KeyGen algorithms, respectively. Table 5.2 presents the comparison of LCP-ABE with the existing schemes with parameters like ciphertext size and secret key size. In Table 5.2, N is the list of the universal attributes, ϕ represents the list of attributes in the access policy, w represents the user attributes, and S_{G_0}, S_{G_1}, and S_{G_T} represents the size of each element in the Groups G_0, G_1, and G_T respectively. B represents the attributes in the access policy.

As shown in Table 5.2, our scheme generates three group elements in ciphertext and two group elements along with the bit string in USK, which is less than existing schemes. Our scheme takes minimal time to communicate the messages. Thus our scheme is more efficient than existing schemes [8, 13, 16]

Table 5.2 Communication overhead

Scheme	Ciphertext size	Secret key size
Hur and Noh [8]	$(2\phi+1).S_{G_0}+S_{G_1}$	$(2w+1).S_{G_0}$
Yu et al. [13]	$(N+1).S_{G_0}+S_{G_T}$	$(2w+1).S_{G_0}$
Naruse et al. [16]	$S_{G_0}+(2\phi+1).S_{G_{T0}}$	$w.S_{G_0}$
LCP-ABE	$2.S_{G_0}+S_{G_T}$	$\lvert B\rvert+2.S_{G_1}$

5.7.2 Computation overhead

Computation overhead depends on expensive operations like exponentiation on the group element and the pairing computations involved in the execution of the encryption, decryption, and keyGen algorithms. Table 5.3 presents the computation costs of the LCP-ABE scheme and existing schemes [8, 13, 16].

Table 5.3 Computation overhead

Scheme	Encrypt	KeyGen	Decrypt
Hur and Noh [8]	$(2\phi + 1).E_G + P_e$	$(2w+ 1).E_G$	$(2w + 1).P_e + \phi_\Psi.E_G$
Yu et al. [13]	$(N + 2).E_G$	$(2w + 1).E_G$	$(N + 1).P_e$
Naruse et al. [16]	$(2\phi + 1).P_e + 2\Phi.\ E_G$	$w.E_G$	$2w.P_e + E_G$
LCP-ABE	$(\lvert\mathrm{WHT}\rvert + 2).E_G$	$2.E_G$	$2.P_e$

In Table 5.3, N represents the universal attributes. E_G represents the number of exponentiation operations performed on the elements of the group, P_e is the number of pairing operations, and ψ represents the number of non-leaf nodes in the access tree.

As shown in Table 5.3, our scheme takes WHT + 2, EG number of operations for the encryption algorithm and two exponentiation operations and two pairing computations required for the KeyGen decryption algorithm, respectively. As such, our scheme takes minimal time to perform all the algorithms.

5.7.3 Experimental results

We performed simulation experiments on a laptop with an Intel i5 2.3 GHz CPU with 8 GB RAM and the Ubuntu 16.04 operating system. We also used an Android 6.0.1 smartphone with quad core processor and 2 GB memory as the IoT device. To implement schemes, we used Python programming language by using Charm-crypto 0.43 Library [23]. Here, we measure the computational cost of our scheme with the help of Figs. 5.3–5.5.

Fig. 5.3 shows the time required to perform the encryption based on different attribute sets. From Fig. 5.3, we can observe that time for encryption increases linearly along with the number of attributes in the access policy.

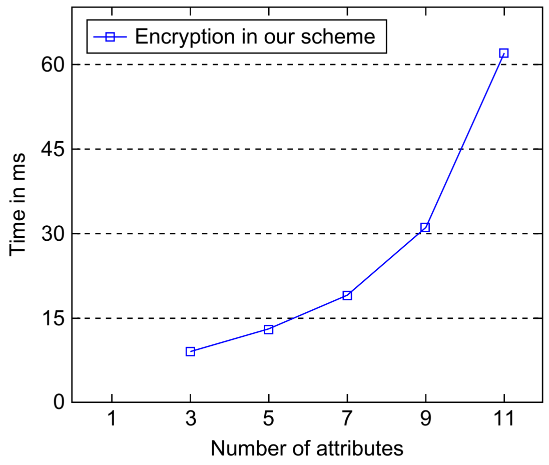


FIG. 5.3

Encryption running time.

Fig. 5.4 shows the time required to perform the KeyGen operation w.r.t to the attributes given by the user. Time for the KeyGen operation changes linearly up to some level and then requires only constant time to generate the USK.

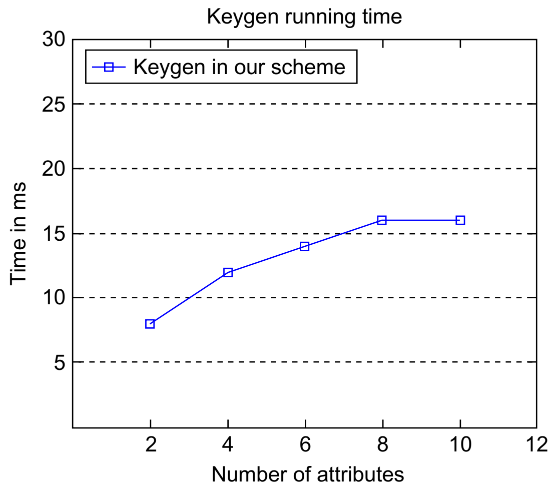


FIG. 5.4

Key generation running time.

Fig. 5.5 shows the time required to perform the decryption operation. Our scheme takes the same amount of time to perform decryption regardless of the number of attributes. It also computes the message in constant time because of the access structure.

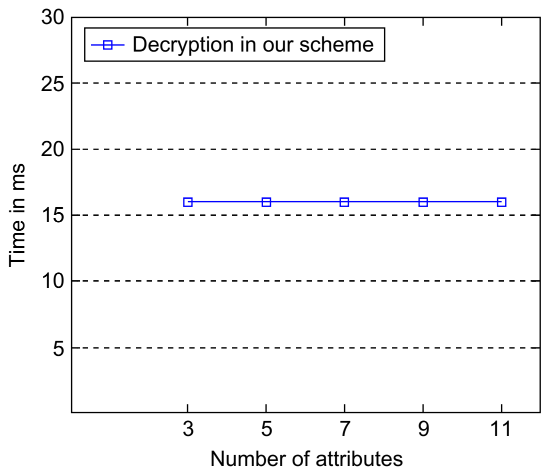


FIG. 5.5

Decryption running time.

5.8 Conclusion

In this chapter, we proposed a *Lightweight CP-ABE* scheme for data privacy and access control in cloud-assisted IoT. We used a new access structure called the Walsh–Hadamard transform that supports user revocation. Our scheme reduced the computation overhead of IoT devices during encryption and allows the occurrence of a single attribute multiple times in access policy during decryption. LCP-ABE proved that it is resistant to collusion attack and chosen plain text attack and maintained forward secrecy. Through performance analysis, we proved that our scheme is lightweight and more suitable for IoT devices. In further work, we extend LCP-ABE to support authentication of IoT devices, integrity of data, and dynamic data operations for IoT data in the cloud.

References

[1] A. Alrawais, A. Alhothaily, C. Hu, X. Cheng, Fog computing for the internet of things: Security and privacy issues. IEEE Inter. Comput. 21 (2) (2017) 34–42, https://doi.org/10.1109/MIC.2017.37.

[2] J. Singh, J. Bacon, H. Ko, T. Pasquier, D. Eyers, Twenty security considerations for cloud-supported IoT. IEEE Internet Things J. 3 (3) (2016) 269–284, https://doi.org/10.1109/JIOT.2015.2460333.

[3] B. Waters, Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization, in: Proceedings of Public Key Cryptography, Vol. 6571, 2011, pp. 53–70.

[4] A. Alrawais, A. Alhothaily, C. Hu, X. Cheng, An attribute-based encryption scheme to secure fog communications. IEEE Access 5 (2017) 9131–9138, https://doi.org/10.1109/ACCESS.2017.2705076.

[5] Bethencourt, J., Sahai, A., and Waters. B.(2007). Ciphertext-policy attribute-based encryption. In Proceedings of the 2007 IEEE Symposium on Security and Privacy, Vol. 24 of SP'07, pages 321–334. IEEE Computer Society, 2007. ISBN 0-7695-2848-1. doi: https://doi.org/10.1109/SP.2007.11.

[6] V. Goyal, R.A. Jain, O. Pandey, A. Sahai, Bounded ciphertext policy attribute based encryption. in: I. Damgård, L.A. Goldberg, M.M. Halldorsson, A. Ingolfsdottir, I. Walukiewicz (Eds.), Automata, Languages and Programming, Springer, Berlin/Heidelberg, 2008, ISBN: 978-3-540-70583-3, pp. 579–591, https://doi.org/10.1007/978-3-540-70583-3n.

[7] Z. Guan, J. Li, L. Wu, Y. Zhang, J. Wu, X. Du, Achieving efficient and secure data acquisition for cloud-supported internet of things in smart grid, IEEE Internet Things J. 4 (6) (2017) 1934–1944, https://doi.org/10.1109/JIOT.2017.2690522.

[8] J. Hur, D.K. Noh, Attribute-based access control with efficient revocation in data outsourcing systems, IEEE Trans. Parallel Distrib. Syst. 22 (7) (2011) 1214–1221, https://doi.org/10.1109/TPDS.2010.203.

[9] Ibraimi, L., Tang, Q., Hartel, P. and Jonker, W.(2009). Efficient and provable secure ciphertext-policy attribute-based encryption schemes. In Proceedings of the 5th International Conference on Information Security Practice and Experience, ISPEC'09, pages 1–12. Springer-Verlag. ISBN 978-3-642-00842-9. doi: https:/doi.org/10.1007/978-3-642-00843-6n.1.

[10] Cheung, L., and Newport, C. Provably secure ciphertext policy abe. In Proceedings of the 14th ACM Conference on Computer and Communications Security, CCS '07, pages 456–465. ACM, 2007. ISBN 978-1-59593-703-2. Doi:https:/doi.org/10.1145/1315245.1315302.

[11] K. Emura, A. Miyaji, K.O. Nomura, M. Soshi, A ciphertext-policy attribute-based encryption scheme with constant ciphertext length, in: F. Bao, H. Li, G. Wang (Eds.), Information Security Practice and Experience, Springer, Berlin/Heidelberg, 2009, ISBN: 978-3-642-00843-6, pp. 13–23.

[12] T. Nishide, K. Yoneyama, K. Ohta, Attribute-based encryption with partially hidden encryptor-specified access structures, in: S.M. Bellovin, R. Gennaro, A. Keromytis, M. Yung (Eds.), Applied Cryptography and Network Security, Springer, Berlin/Heidelberg, 2008, ISBN: 978-3-540-68914-0, pp. 111–129.

[13] Yu, S., Wang, C., Ren, K., and Lou, W (2010). Attribute based data sharing with attribute revocation. In Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security, ASIACCS '10, pages 261–270. ACM, ISBN 978-1-60558-936-7. doi: https:/doi.org/10.1145/1755688.1755720.

[14] Hu, C., Zhang, F., Xiang, T., Li, H., Xiao, X. and Huang, G.(2014). A practically optimized implementation of attribute based cryptosystems. In 2014 IEEE 13th International Conference on Trust, Security and Privacy in Computing and Communications, Pages 197–204, doi: https:/doi.org/10.1109/TrustCom.2014.29 (web archive link).

[15] Liu, Z., and Cao, Z. (2010), On efficiently transferring the linear secret-sharing scheme matrix in ciphertext-policy attribute-based encryption. 2010:374

[16] T. Naruse, M. Mohri, Y. Shiraishi, Attribute-based encryption with attribute revocation and Grant function using proxy re-encryption and attribute key for updating, in: J. Park, I. Stojmenovic, M. Choi, F. Xhafa (Eds.), Future Information Technology. Lecture Notes in Electrical Engineering, Vol. 276, Springer, Berlin/Heidelberg, 2014.

[17] P. Kumar, S. Kumar, A.P.J. Alphonse, Attribute based encryption in cloud computing. J. Netw. Comput. Appl. 1084-8045108 (C) (2018) 37–52, https://doi.org/10.1016/j.jnca.2018.02.009.

[18] A. Shamir, How to share a secret, Commun. ACM 0001-078222 (11) (1979) 612–613, https://doi.org/10.1145/359168.359176.

[19] F. Guo, Y. Mu, W. Susilo, D.S. Wong, V. Varadharajan, Cp-abe with constant-size keys for lightweight devices, IEEE Trans. Inf. Forensics Security 1556-60139 (5) (2014) 763–771, https://doi.org/10.1109/TIFS.2014.2309858.

[20] Lewko, A. B. Okamoto, T., Sahai, A., Takashima, K., and Waters, B. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption", Proceedings of EUROCRYPT, 2010, Vol. 6110, pp. 62–91.

[21] A. Lewko, B. Waters, New proof methods for attribute based encryption: Achieving full security through selective techniques, in: 32nd International Conference on Cryptology (CRYPTO 2012), Springer, Santa Barbara, 2012, , pp. 180–198.

[22] C.-K. Wu, D. Feng, Boolean Functions and Their Walsh Transforms, Springer, Berlin/Heidelberg, 2016, ISBN: 978-3-662-48865-2, pp. 1–30, https://doi.org/10.1007/978-3-662-48865-2n.1.

[23] A. Akinyele, C. Garman, I. Miers, M.W. Pagano, M. Rushanan, Green, and A. D. Rubin. Charm: a framework for rapidly prototyping cryptosystems, J. Cryptog. Eng. 2190-85163 (2) (2013) 111–128, https://doi.org/10.1007/s13389-013-0057-3.

[24] B.J. Fino, V.R. Algazi, Unified matrix treatment of the fast walsh-hadamard transform, IEEE Trans. Comput. 0018-934025 (11) (1976) 1142–1146, https://doi.org/10.1109/TC.1976.1674569.