On Anonymous Attribute Based Encryption

Payal Chaudhari[1,2], Manik Lal Das[2](✉), and Anish Mathuria[2]

[1] LDRP-ITR, Gandhinagar, India
payal.ldrp@gmail.com
[2] DA-IICT, Gandhinagar, India
{maniklal_das,anish_mathuria}@daiict.ac.in

Abstract. Attribute Based Encryption (ABE) has found enormous scope in data confidentiality and fine-grained access control of shared data stored in public cloud. Classical ABE schemes require attaching the access policy along with the ciphertext, where the access policy describes required attribute values of a receiver. As attributes of a receiver (i.e., user) could relate to the identity of users, it could lead to reveal some sensitive information of the ciphertext (e.g. nature of plaintext, action sought from of receiver) for applications like healthcare, financial contract, bureaucracy, etc. Therefore, anonymizing attributes while sending ciphertext in use of ABE schemes, known as Anonymous ABE (AABE), is a promising primitive for enforcing fine-grained access control as well as preserving privacy of the receiver. In ASIACCS 2013, Zhang *et al.* proposed an AABE scheme using the *match-then-decrypt* [1] technique, where before performing decryption, the user performs a match operation that ensures a user whether he is the intended recipient for the ciphertext or not. We found that Zhang *et al.*'s scheme [1] is not secure, in particular, it fails to achieve receiver's anonymity. In this paper, we discuss the security weaknesses of Zhang *et al.*'s scheme. We show that an adversary can successfully check whether an attribute is required to decrypt a ciphertext, in turn, reveal the receiver's identity. We also suggest an improved scheme to overcome the security weakness of Zhang *et al.*'s scheme.

Keywords: Attribute based encryption · Anonymity · Bilinear pairing · Access structure

1 Introduction

Cloud infrastructure provides important features to service providers and consumers such as high data availability, reliability and low-cost maintainability of stored data in cloud server. While storing data in third party cloud server and accessing it over public channel security of users data and privacy of data access are become an active research problem in recent times. Attribute Based Encryption (ABE) [2–4] is a public-key cryptographic primitive suitable for data confidentiality and fine-grained access control enforced in public cloud. ABE is more flexible than conventional public-key encryption, as ABE supports one-to-many encryption instead of one-to-one. With ABE, a data owner can share the data with multiple designated users by sending ciphertext, pertaining to target user's attributes. There are two kinds of ABE – (1) Key-Policy Attribute Based Encryption(KP-ABE) [2,4]; and (2) Ciphertext-Policy Attribute Based Encryption (CP-ABE) [3]. In KP-ABE, each ciphertext is labeled by the encryptor with a set of descriptive attributes and the private key of a user is associated with an access structure that specifies which type of ciphertext the user can decrypt. Whereas, in CP-ABE a user is identified by a set of attributes which are included in his private key, and a data owner can decide the access policy for decrypting ciphertext intended to the user. The encrypted message must specify an associated access policy over attributes. A user can only able to decrypt a ciphertext if the user's attributes pass through the ciphertext's access policy.

Although ABE scheme supports fine-grained access control, it discloses receiver's identity by which an adversary can guess the purpose of the message from the ciphertext by seeing receiver's attributes. For example, the adversary can guess that the receiver is a faculty if some of the attributes are *question paper*, *student*, *first year*, *discipline*, etc. Therefore, protecting receiver's identity while using ABE is a challenging research problem.

Anonymous ABE (AABE) is introduced in [5–8] as a promising public-key primitive that allows sender in achieving receiver anonymity in ABE. In anonymous CP-ABE, access policy is hidden in the ciphertext. A user requires to decrypt a ciphertext using secret key belongs to his attributes. If his secret key matches with the access policy, the user can successfully decrypt the ciphertext. If the attribute set associated with the secret key does not match with the access policy, then the user cannot get what access policy is specified by the encryptor. Therefore, the user in AABE schemes is required to perform the whole decryption procedure in order to verify if he is the intended receiver of the ciphertext or not, which results into a large overhead on the user when the ciphertext is not intended to him, but the user is engaged with the decryption procedure for the ciphertext.

In ASIACCS 2013, Zhang *et al.* [1] proposed an AABE scheme to address receiver anonymity by adding one matching phase before decryption of the ciphertext. The user performs the match-then-decrypt procedure using his secret key components and ciphertext components to check if he is the intended recipient of the ciphertext. The scheme of Zhang *et al.* is efficient than other AABE schemes in the sense that all receiver do not engage in full decryption procedure used in other schemes, instead after the partial decryption (i.e., match-then-decrypt phase) the intended receiver goes for the final decryption procedure in Zhang *et al.*' scheme. However, we found that Zhang *et al.*'s scheme is not secure, that is, it does not support receiver's anonymity. Any user of the system or an outsider (say, adversary) can successfully check whether an attribute is required to decrypt a ciphertext, in turn, reveal receiver's identity.

In this paper, we show the security weaknesses of Zhang *et al.*'s scheme. We propose an improved scheme to mitigate the security weakness of

Zhang *et al.*'s scheme. We show that the improved scheme is secure with respect to the security claim of the Zhang *et al.*'s scheme.

The remainder of the paper is organized as follows. Section 2 gives some preliminaries. Section 3 reviews Zhang *et al.*'s scheme [1]. Section 4 discusses the security flaws of Zhang *et al.*'s scheme [1]. Section 5 presents the improved scheme followed by its analysis in Sect. 6. Section 7 provides the performance analysis of the improved scheme. We conclude the paper in Sect. 8.

2 Preliminaries

In order to make the paper self-contained, we provide some preliminaries that have been used throughout the paper.

2.1 Bilinear Mapping

Let G and G_T be two multiplicative cyclic groups of prime order p. Let g be a generator of G and e be a bilinear map, $e : G \times G \rightarrow G_T$. The bilinear map e has the following properties:

- Bilinearity: for all $u, v \in G$ and a, $b \in \mathbb{Z}_p$, we have $e(u^a, v^b) = e(u, v)^{ab}$.
- Non-degeneracy: $e(g, g) \neq 1$.
- e: $G \times G \rightarrow G_T$ is efficiently computable.

2.2 Complexity Assumption

Decisional Linear (D-Linear) Assumption. Let z_1, z_2, z_3, z_4, $z \in \mathbb{Z}_p$ be chosen at random and g be a generator a cyclic group G. The decisional Linear assumption [9] is that no probabilistic polynomial-time algorithm $\mathcal{P}$ can distinguish the tuple ($Z_1 = g^{z_1}$, $Z_2 = g^{z_2}$, $Z_3 = g^{z_1 z_3}$, $Z_4 = g^{z_2 z_4}$, $Z = g^{z_3+z_4}$) from the tuple ($Z_1 = g^{z_1}$, $Z_2 = g^{z_2}$, $Z_3 = g^{z_1 z_3}$, $Z_4 = g^{z_2 z_4}$, $Z = g^z$) in G with more than a negligible advantage ϵ.

The advantage of $\mathcal{P}$ is Pr[$\mathcal{P}$ (Z_1, Z_2, Z_3, $g^{z_3+z_4}$) = 0] - Pr[$\mathcal{P}$(Z_1, Z_2, Z_3, g^z) = 0] = ϵ where the probability is taken over the random choice of the generator g, the random choice of z_1, z_2, z_3, z_4, $z \in \mathbb{Z}_p$, and the random bits consumed by $\mathcal{P}$.

For the proof of our proposed improved scheme we consider a variant of D-Linear Assumption [7] which states that no probabilistic polynomial-time algorithm $\mathcal{P}$ can distinguish the tuple ($Z_1 = g^{z_1}$, $Z_2 = g^{z_2}$, $Z_3 = g^{z_3+z_4}$, $Z_4 = g^{z_2 z_4}$, $Z = g^{z_1 z_3}$) from the tuple ($Z_1 = g^{z_1}$, $Z_2 = g^{z_2}$, $Z_3 = g^{z_3+z_4}$, $Z_4 = g^{z_2 z_4}$, $Z = g^z$) in G with more than a negligible advantage ϵ.

Decisional Diffie-Hellman (DDH) Assumption. Let a,b,$z \in \mathbb{Z}_p$ be chosen at random and g be a generator of a cyclic group G. The decisional Diffie Hellman assumption is that no probabilistic polynomial-time algorithm $\mathcal{B}$ can distinguish the tuple $(g, P = g^a, Q = g^b, R = g^{ab})$ from the tuple $(g, P = g^a, Q = g^b, R = g^z)$ with more than a negligible advantage ϵ.

The advantage of $\mathcal{P}$ is Pr[$\mathcal{P}$(g,g^a,g^b,g^{ab}) = 0] - Pr[$\mathcal{P}$(g,g^a,g^b,g^z) = 0] = ϵ where the probability is taken over the random choice of the generator g, the random choice of *a, b, z* $\in \mathbb{Z}_p$, and the random bits consumed by $\mathcal{P}$.

2.3 Access Structure

Let there be n attributes in the universe and each attribute i (for all $1 \leq i \leq n$) has value set $V_i = \{v_{i,1}, v_{i,2}, \cdots, v_{i,n_i}\}$. $L = [L_1, \mathrm{L}_2, \cdots, \mathrm{L}_n]$ is an attribute list, where each L_i represents one value from the value set of attribute i. A ciphertext policy $W = [W_1, W_2, \cdots, W_n]$ where $W_i \subseteq V_i$ for $1 \leq i \leq n$. Each W_i represents the set of permissible values of an attribute i in order to decrypt the ciphertext. An access structure W is a rule that returns 1 when given a set L of attributes if L satisfies W, else, it returns 0. An attribute list L satisfies W, if $L_i \in W_i$ for all $1 \leq i \leq n$.

3 Zhang *et al.'s* Scheme

3.1 Scheme Definition

Zhang *et al.*'s scheme [1] consists of four algorithms – **Setup, KeyGen, Encrypt**, and **Decrypt**, which are defined as follows:

- Setup(1^l) $\rightarrow$ (PK,MK): The setup algorithm is run by the attribute center. On input a security parameter l it returns public key PK which is distributed to users, and the master key MK which is kept private.
- KeyGen(PK,MK,L) $\rightarrow SK_L$: This algorithm is run by the attribute center. On input the public key PK, the master key MK and an attribute List L, it outputs SK_L as the attribute secret key associated with the attribute list L.
- Encrypt(PK, M, W) $\rightarrow CT_W$: An encryptor runs this probabilistic algorithm. The input to the algorithm is public key PK, a message M, and a ciphertext policy W, and output is a ciphertext CT_W which is a encryption of M with respect to W.
- Decrypt(PK, CT_W, SK_L) $\rightarrow M$ or $\perp$: The decryption algorithm is deterministic and it involves two phases, attribute matching detection and decryption phase. When user provides as input the system public key PK, a ciphertext CT_W and a secret key SK_L associated with L, the algorithm proceeds as follows:
  1. Matching Phase: If the attribute list L associated with SK_L matches with the ciphertext policy W of CT_W then it initiates Decryption phase, else, it returns $\perp$ and terminates decryption.
  2. Decryption Phase: It returns message M.

3.2 Detailed Construction

- **Setup**(1^l): Let G, G_T be cyclic multiplicative groups of prime order p, and $e : G \times G \rightarrow G_T$ be a bilinear map. $H : \{0,1\}^* \rightarrow G$ is a map-to-point function that takes a string as input and outputs a point on elliptic curve. The attribute center chooses $y \in_R \mathbb{Z}_p, g, g_1, g_2 \in_R G$, and computes $Y = e(g_1, g_2)^y$. The system public key is $PK = \langle g, g_1, g_2, Y \rangle$, and the master key is $\langle y \rangle$.

- **KeyGen**(PK,MK,L): Let $L = [L_1, L_2, \cdots, L_n]$ be the attribute list for the user who requires a secret key. The attribute center chooses $r_1, r_2, \cdots, r_{n-1} \in_R \mathbb{Z}_p$ and computes $r_n = y - \sum_{i=1}^{n-1} r_i \pmod p$. Then the attribute center chooses $r \in_R \mathbb{Z}_p$ and $\{\hat{r_i}, \lambda_i, \hat{\lambda_i} \in_R \mathbb{Z}_p\}_{1 \le i \le n}$, sets $\hat{r} = \sum_{i=1}^{n} \hat{r_i}$ and computes $[\hat{D_0}, D_{\Delta,0}] = [g_2^{y-\hat{r}}, g_1^r]$. For $1 \le i \le n$, the attribute center computes $[D_{\Delta,i}, D_{i,0}, D_{i,1}, \hat{D}_{i,0}, \hat{D}_{i,1}] = [g_2^{\hat{r_i}} H(i||v_{i,k_i})^r, g_2^{\lambda_i}, g_1^{r_i} H(0||i||v_{i,k_i})^{\lambda_i}, g_1^{\hat{\lambda_i}}, g_2^{r_i} H(1||i||v_{i,k_i})^{\hat{\lambda_i}}]$ where $L_i = v_{i,k_i}$.
  The secret key is $SK_L = \langle \hat{D_0}, D_{\Delta,0}, \{D_{\Delta,i}, D_{i,0}, D_{i,1}, \hat{D}_{i,0}, \hat{D}_{i,1}\}_{1 \le i \le n} \rangle$.
- **Encrypt**(PK,M,W): For encryption of a message M with respect to access control policy W, encryptor selects s, s' and, $s'' \in_R \mathbb{Z}_p$ and computes $\tilde{C} = MY^s, C_\Delta = e(g,g)^s Y^{s'}, C_0 = g^s, \hat{C}_0 = g_1^{s'}, C_1 = g_2^{s''}, \hat{C}_1 = g_1^{s-s''}$. Then for $1 \le i \le n$ and $1 \le j \le n_i$ the encryptor computes $[C_{i,j,\Delta}, C_{i,j,0}, \hat{C}_{i,j,0}]$ as follows: If $v_{i,j} \in W_i$ then $[C_{i,j,\Delta}, C_{i,j,0}, \hat{C}_{i,j,0}] = [H(i||v_{i,j})^{s'}, H(0||i||v_{i,j})^{s''}, H(1|| i||v_{i,j})^{s-s''}]$ else if $v_{i,j} \notin W_i$ then $[C_{i,j,\Delta}, C_{i,j,0}, \hat{C}_{i,j,0}]$ are random elements.
  The encryptor prepares $CT_W = \langle\ C_\Delta, C_0, \hat{C}_0, \tilde{C}, C_1, \hat{C}_1, \{\{C_{i,j,\Delta}, C_{i,j,0}, \hat{C}_{i,j,0}\}_{1 \le j \le n_i}\}_{1 \le i \le n} \rangle$.
- **Decrypt**(PK, CT_W, SK_L): A receiver of the ciphertext tests and decrypts the ciphertext CT_W using his secret key SK_L as follows:

  1. Matching Phase: Receiver checks if his attributes L satisfies W or not by checking if following equality holds true. In the following equation the $C_{i,j}$ denotes the cipher component related to j^{th} value of an attribute i which a receiver possesses.
  $$\frac{C_\Delta}{e(g, C_0)} = \frac{e(\hat{C}_0, \hat{D}_0 \prod_{i=1}^{n} D_{\Delta,i})}{e(\prod_{i=1}^{n} C_{i,j,\Delta}, D_{\Delta,0})}$$
  If the equality holds false then decryption procedure is aborted; otherwise, the Decryption Phase is initiated.
  2. Decryption Phase: The receiver recovers message M using following computation
  $$M = \frac{\tilde{C} \prod_{i=1}^{n} e(C_{i,j,0}, D_{i,0}) e(\hat{C}_{i,j,0}, \hat{D}_{i,0})}{\prod_{i=1}^{n} e(C_1, D_{i,1}) e(\hat{C}_1, \hat{D}_{i,1})}.$$

4 Security Flaws in Zhang *et al.'s* Scheme

The authors of the scheme [1] proposed a cost-effective decryption procedure with a matching phase operation before the decryption procedure. The authors in [1] claimed that the scheme provides anonymity, and the ciphertext does not disclose the identity of the receiver. They have also stated that if any receiver succeeds in decryption of a message, he is not be able to identify who else can decrypt the same ciphertext. However, the authors did not provide the security proof for the matching phase elements and match operation. The security proof presented in their scheme primarily focused on the matter that the ciphertext is not distinguishable from any other random element, if the adversary does not possess the corresponding attribute secret key. We found that the cipher components for matching phase themselves discloses the underlying ciphertext access policy. In this section we show that the scheme [1] does not provide receiver's anonymity

We consider as an adversary, any user inside the system or any outsider, who has knowledge of universe of attributes. The adversary can successfully check if a particular attribute is included in ciphertext. In particular, the attributes which make the attack successful are $\hat{C}_0$ and $\{\{C_{i,j,\Delta}\}_{1\leq j\leq n_i}\}_{1\leq i\leq n}$. To check whether an attribute $v_{i,j}$ is included in ciphertext or not, the adversary calculates $D'_{\Delta,i,j} = H(i||v_{i,j})$. Then, the adversary checks if following equation returns true for an attribute $\mathrm{v}_{i,j}$.

$$e(\hat{C}_0, D'_{\Delta,i,j}) = e(C_{i,j,\Delta}, g_1)$$

If the above equality holds true, the adversary can conclude that the attribute used in the equation is included in ciphertext access policy. With this, the adversary now checks if a specific attribute which may be an identity of a user is integrated in access policy or not.

For example, suppose a University has three different departments *"Computer Science, Electrical Engineering, and Mechanical Engineering"*. The attribute categories and their corresponding value sets are as follows.

- For the attribute "*Role*" $W_{Role} = \{Dean, Teacher, Student, Administrative\ Staff\}$
- For the attribute "*Department*" $W_{Dept} = \{CS, EL, ME\}$
- For the attribute "*Course*" $W_{Course} = \{PhD, MS, BS\}$

When the Dean sends some confidential notice to all teachers in an encrypted form using the scheme [1], the Dean generates an encrypted message with following ciphertext components. For simplicity, we do not show all ciphertext components. Instead, we provide the ciphertext components for the attribute "*Role*".

$$\begin{aligned}
C_\Delta &= e(g,g)^s Y^{s'} \\
C_0 &= g^s \\
\hat{C}_0 &= g_1^{s'} \\
\tilde{C} &= MY^s \\
C_1 &= g_2^{s''} \\
\hat{C}_1 &= g_1^{s-s''}
\end{aligned}$$

$$\begin{aligned}
&\{C_{Role,Teacher,\Delta}, C_{Role,Teacher,0}, \hat{C}_{Role,Teacher,0}\} \\
&= \{H(Role\|Teacher)^{s'}, H(0\|Role\|Teacher)^{s''}, H(1\|Role\|Teacher)^{s-s''}\}
\end{aligned}$$

Random values are provided for other attributes such as *Student*, *Dean* and *Administrative staff*.

The adversary now checks whether a *Teacher* is the intended recipient of the ciphertext with following equation.

$$e(\hat{C}_0, H(Role\|Teacher)) = e(C_{Role,Teacher,\Delta}, g_1).$$

The correctness of the equation is shown below.

$$\begin{aligned} & e(\hat{C}_0, H(Role\|Teacher)) \\ = & e(g_1^{s'}, H(Role\|Teacher)) \\ = & e(H(Role\|Teacher)^{s'}, g_1) \\ = & e(C_{Role,Teacher,\Delta}, g_1) \end{aligned}$$

To recover the whole access policy the adversary requires $n_i \times n$ bilinear pairing operations. Let $m = \max\ (n_i)_{1 \leq i \leq n}$. Therefore, to disclose the receiver's identity the adversary requires at most *O(mn)* bilinear pairing operations.

5 An Improved Scheme

The security weakness of the scheme in [1] occurs in the *matching phase*. The authors in [1] do not provide any security proof for the matching phase components. In the improved scheme we provide an improved matching phase that can be incorporated with any existing Anonymous ABE schemes. We do not include the cipher components and key components required for encryption and decryption of a message as it depends on the AABE scheme used for encryption and decryption of a message. The proposed modified scheme uses a set of parameters which is isolated from the parameters used for the encryption and decryption of a message. The underlying access structure for the improved scheme is as described in Sect. 2.3.

5.1 Scheme Definition

Like Zhang *et al.*'s scheme [1], the improved scheme consists of Setup, KeyGen, Encrypt, and Decrypt phases, which are defined as follows.

Setup($\mathbf{1}^l$): The Setup algorithm takes as input a security parameter *l*. The output of this phase is Master Secret Key *MSK* and Master Public Key *MPK*.

KeyGen(*MSK*,*MPK*,*L*): On input of an attribute list L, MSK and MPK, the algorithm outputs user's secret key SK_L.

Encrypt(*MPK*, *W*): The Encrypt algorithm takes as input the ciphertext access policy W required for decryption and *MPK*. The output of this algorithm is cipher components CT_W, which are used for matching phase.

Match(*MPK*,$\boldsymbol{SK_L}$,$\boldsymbol{CT_W}$): Match phase enables the receiver to check whether he is the intended receiver or not. It takes as input *MPK*, SK_L, CT_W, and returns whether the SK_L is matched with CT_W or not. The output of the algorithm will guide the receiver as he should perform the decryption of the received ciphertext or not.

5.2 Detailed Construction

The scheme works as follows.

- **Setup**(1^l): The Attribute Center (AC) performs the setup phase. It selects two groups G and G_T of prime order p whose bit-length is l and a bilinear mapping function $e : G \times G \rightarrow G_T$. The AC chooses two random generators g_1 and g_2 from group G and one hash function H is defined as H: $\{0,1\}^* \rightarrow G$. The master secret key MSK is chosen as $\langle \alpha, \beta \in_R \mathbb{Z}_p \rangle$. The corresponding master public key MPK $\left\langle g_1, g_2, g_1^{\alpha}, g_2^{\beta}, e(g_1,g_2)^{\alpha} \right\rangle$ is published.
- **KeyGen**(MSK,MPK,L): Let L=$[L_1, L_2, \cdots, L_n]$ be the attribute list for the user who requires a secret key. Here L_i represents a value $v_{i,j}$ that a user possess for attribute i. A user possess exactly one value $v_{i,j}$ for each attribute i where $1 \leq i \leq n$. For every user in the system the AC picks a random value ρ and generates a user's secret key SK_L for performing the matching phase operation as follows.

$$D = (g_2 \prod\nolimits_{i=1}^{n} H(i \| v_{i,j}))^{\alpha} \cdot g_2^{\rho} \text{ where } L_i = v_{i,j}.$$
$$\bar{D} = g_1^{\frac{\rho}{\beta}}$$

- **Encrypt**(MPK, W): We provide the construction of cipher components for matching phase only. Therefore, we have not included encryption of message. The algorithm takes the access policy W and public key MPK as input. Here, W =$\{W_1, W_2, \cdots W_n\}$ where W_i $\{1 \leq i \leq n\}$ is the set of values permissible for decryption. To prepare the cipher components for matching phase the encryptor takes secret values s and t from $\mathbb{Z}_p$ and makes n portions of t as t_i such that $\sum_{i=1}^{n} t_i = t$. For attribute values from each set W_i create the following cipher components.
  - If $v_{i,j} \in W_i$ $\tilde{C_{i,j}} = g_2^{t_i} H(i \| v_{i,j})^{st}$
  - If $v_{i,j} \notin W_i$ $\tilde{C_{i,j}}$ is a random value.

  The other cipher components are $\hat{C} = g_1^{st}$,$\bar{C} = g_2^{st\beta}$ and $C' = e(g_1, g_2)^{\alpha(s-1)t}$.
- **Match**(MPK, SK_L, CT_W): A user performs the match operation before going for decryption of an encrypted message. The user checks if his set of attribute values L satisfies access policy W or not by checking if following equality holds true. User collects the relevant $\tilde{C}_{i,j}$ cipher components. Here $\tilde{C}_{i,j}$ denotes cipher component related to value $v_{i,j}$ for an attribute i which a receiver possesses.

$$C' = \frac{e(\hat{C}, D)}{e(\prod_{i=1}^{n} \tilde{C}_{i,j}, g_1^{\alpha}) e(\bar{C}, \bar{D})}$$

  If the equality does not hold true, the decryption procedure for a message is aborted, otherwise, the user initiates a decryption procedure related to encryption scheme used for encrypting a message.

Correctness: The correctness of the matching phase is as follows.

$$\begin{aligned}
&\frac{e(\hat{C}, D)}{e(\prod_{i=1}^{n} \tilde{C}_{i,j}, g_1^{\alpha})e(\bar{C}, \bar{D})} \\
&= \frac{e(g_1^{st}, (g_2 \prod_{i=1}^{n} H(i||v_{i,j}))^{\alpha} \cdot g_2^{\rho})}{e(\prod_{i=1}^{n} (g_2^{t_i} H(i||v_{i,j})^{st}), g_1^{\alpha})e(g_2^{st\beta}, g_1^{\frac{\rho}{\beta}})} \\
&= e(g_1, g_2)^{(\alpha(s-1))t} \\
&= C'
\end{aligned}$$

After the matching phase, the receiver goes for the final decryption procedure as in [1] to obtain the message.

6 Security Analysis

6.1 Security Model

We consider the IND-sCP (Indistinguishability against selective ciphertext policy) model to analyze the proposed improved scheme. In the analysis we exclude the encryption and decryption of a message as they are not part of the proposed matching scheme. The improved scheme is simulated with the following security game.

Init: The adversary $\mathcal{A}$ commits two ciphertext Policies W_0^* and W_1^* that he wishes to be challenged upon.

Setup: The challenger $\mathcal{B}$ chooses l as a security parameter and chooses α at random from $\mathbb{Z}_p$. $\mathcal{B}$ also defines a bilinear mapping function from $G \times G \rightarrow G_T$ and chooses two generators from G as g_1, g_2. The master private key is α. The public parameters g_1, g_2, g_1^{α}, g_2^{β} and $e(g_1, g_2)^{\alpha}$ are sent to $\mathcal{A}$.

The game has following four steps.

Step 1. **Preprocessing Phase.** With this phase $\mathcal{A}$ issues polynomially bounded number of queries and gathers following items from the challenger.
- Secret key SK_L for attribute set L such that L satisfies either both challenge ciphertext policies W_0^* and W_1^* or satisfies none of them.
- matching phase elements for different access policies W.

Step 2. **Challenge Phase.** The challenger $\mathcal{C}$ randomly picks a bit $\nu = 0$ or 1 and submits the ciphertext elements for matching phase related to W_{ν}^* using two random secret values s, t from $\mathbb{Z}_p$.

Step 3. **Post Processing Phase.** The adversary $\mathcal{A}$ is allowed to run a number of queries as done in preprocessing phase.

Step 4. **Guess**: The adversary $\mathcal{A}$ outputs a guess ν'. $\mathcal{A}$ wins the game if $\nu' = \nu$. The advantage of $\mathcal{A}$ in this game is defined as $Adv_{\mathcal{A}}(1) = |Pr[\nu' = \nu] - 1/2|$.

We show that the improved scheme is secure in the IND-sCP model. We prove the security of the scheme relies on the hardness of D-Linear Assumption and Decisional Diffie-Hellman assumption. We prove the security of proposed scheme in two theorems. In first theorem we prove that unless a valid decryption key is available, the adversary is not able to find a valid match. In the second theorem we prove that receiver anonymity is preserved in the improved scheme. We show that even if an adversary is able to get a valid key and find the match successfully, he can not find out the underlying access policy.

The security model consists of a Challenger $\mathcal{C}$, a Simulator $\mathcal{S}$ and an Adversary $\mathcal{A}$.

Theorem 1. *If an adversary can break the proposed improved scheme in the random oracle model, then a simulator can be constructed who can break the D-Linear assumption with a non-negligible advantage.*

Proof. We show that without a correct decryption key $\mathcal{A}$ is not able to compute any function of C'. If an adversary is able to succeed in doing so with non-negligible advantage ϵ_1, then we are able to design a simulator $\mathcal{S}$ that can play the D-Linear game with advantage $\frac{\epsilon_1}{2}$. For the proof we consider a variant of D-Linear assumption. The simulation proceeds as follows: We first let the challenger set the groups G and G_T of prime order p, with an efficient bilinear map, e and generator g. The challenger flip a fair binary coin μ, outside of $\mathcal{S}$'s view. If $\mu = 0$, the challenger sets $(g, Z_1, Z_2, Z_3, Z_4, Z) = (g, g^{z_1}, g^{z_2}, g^{z_2 z_4}, g^{z_3+z_4}, g^{z_1 z_3})$, otherwise it sets $(g, Z_1, Z_2, Z_3, Z_4, Z) = (g, g^{z_1}, g^{z_2}, g^{z_2 z_4}, g^{z_3+z_4}, g^{z})$ for values z_1, z_2, z_3, z_4 and z chosen randomly from $\mathbb{Z}_p$.

Init: The simulator $\mathcal{S}$ runs $\mathcal{A}$. $\mathcal{A}$ commits two access policies W_0^* and W_1^* for which he wishes to be challenged upon.

Setup: $\mathcal{S}$ takes the following values: $g_1 = g^a$, $g_2 = g$, $g_1^\alpha = g^{a\alpha}$ with the assumption that $\alpha = z_1$. Here a is chosen randomly from $\mathbb{Z}_p$. With the selection of random value β from $\mathbb{Z}_p$, g_2^β is calculated as g^β. $H(i \| v_{i,j})$ is assumed as $g^{\frac{1}{nz_1}+H'(i\|v_{i,j})}$. Here n denotes the number of attribute categories and $H'(i\|v_{i,j})$ is computed from random oracle, producing an element of $\mathbb{Z}_p$ in output. The simulator $\mathcal{S}$ announces the public key as $g_1 = g^a$, $g_2 = g$, $g_1^\alpha = g^{az_1}$, $g_2^\beta = g^\beta$, $\mathrm{e}(\mathrm{g}_1,\mathrm{g}_2)^\alpha = \mathrm{e}(\mathrm{g},\mathrm{g})^{az_1}$.

Preprocessing Phase: The adversary $\mathcal{A}$ gathers following information from the simulator $\mathcal{S}$.

– Whenever $\mathcal{A}$ makes its k^{th} key generation query for the set S_k of attributes such that S_k satisfies neither W_0^* nor W_1^*. The simulator $\mathcal{S}$ selects a random value $\rho \in \mathbb{Z}_p$ and the resultant key components are generated as follows.

$$\begin{aligned} D &= (g_2 \prod_{i=1}^{n} H(i\|v_{i,j_i}))^\alpha \cdot g_2^\rho \\ &= (g \prod_{i=1}^{n} g^{\frac{1}{nz_1}+H'(i\|v_{i,j_i})})^{z_1} \cdot g^\rho \end{aligned}$$

$$
\begin{aligned}
&= g^{z_1} \cdot g^{1+\sum_{i=1}^{n}(H'(i\|v_{i,j_i})z_1)} \cdot g^{\rho} \\
&= Z_1 \cdot g \cdot Z_1^{\sum_{i=1}^{n}(H'(i\|v_{i,j_i}))} \cdot g^{\rho} \\
\bar{D} &= g_1^{\frac{\rho}{\beta}} = g^{\frac{a\rho}{\beta}}
\end{aligned}
$$

– When $\mathcal{A}$ issues queries for ciphertext elements related to any access policy W, then the simulator $\mathcal{S}$ chooses random values s and t from $\mathbb{Z}_p$ and provides the output of matching phase components incorporating access policy W.

Challenge: Let the two challenge ciphertext policies submitted by the adversary $\mathcal{A}$ are $W_0^* = [W_{0,1}, W_{0,2}, \cdots, W_{0,n}]$ and $W_1^* = [W_{1,1}, W_{1,2}, \cdots, W_{1,n}]$.

Now $\mathcal{S}$ does following. $\mathcal{S}$ flips a random coin ν, and computes the cipher components for W_ν^* as follows. $\mathcal{S}$ assumes st=z_1z_3, with the assumptions that $s = \frac{z_1z_3}{z_4(z_2+1)}$ and $t = z_4(z_2+1)$. For the values which are included in W_ν^*, $\mathcal{C}$ calculates $\tilde{C}_{i,j} = g_2^{t_i} H(i\|v_{i,j_i})^{st} = g^{\frac{z_2z_4+z_4}{n}} g^{(\frac{1}{nz_1}+H'(i\|v_{i,j_i}))z_1z_3} = Z_4^{\frac{1}{n}} Z_3^{\frac{1}{n}} Z^{H'(i\|v_{i,j_i})}$.

For other attribute values which are not included in W_ν^*, $\tilde{C}_{i,j}$ are random values. Subsequent cipher components are calculated by $\mathcal{S}$ as $\hat{C} = g_1^{st} = Z^a$, $\bar{C} = g_2^{st\beta} = Z^{\beta}$ and $C' = \frac{e(Z_1^a,Z)e(g,Z)}{e(Z_1^a,Z_4)e(Z_1,Z_3)}$. C' is correct cipher component only if $Z = g^{z_1z_3}$. Else C' is a random element. Ciphertext components $\tilde{C}, \bar{C}, \hat{C}, C'$ are given to $\mathcal{A}$.

Post Processing Phase: $\mathcal{A}$ is allowed to run a number of queries for attribute keys and ciphertext components for matching phase with the same conditions as imposed in the preprocessing phase.

Guess: $\mathcal{A}$ submits a guess ν' of ν. If $\nu' = \nu$, then $\mathcal{S}$ outputs μ=1 to indicate that it was given a valid D-Linear tuple, else it outputs μ=0 to indicate that the ciphertext is a random element. Therefore, $\mathcal{A}$ gains no information about ν, in turn, $Pr[\nu \neq \nu'|\mu = 0]= \frac{1}{2}$. As the simulator guesses μ'=0 when $\nu \neq \nu'$, $Pr[\mu = \mu'|\mu = 0] = \frac{1}{2}$. If $\mu = 1$, then the adversary $\mathcal{A}$ is able to view a valid matching phase components with advantage $\epsilon_1(l)$, a negligible quantity in security parameter in l. Therefore, $Pr[\nu = \nu'|\nu = 1] = \frac{1}{2} + \epsilon_1(l)$. Similarly, the simulator $\mathcal{S}$ guesses μ'=1 when $\nu = \nu'$, in turn, $Pr[\mu' = \mu|\mu = 1] = \frac{1}{2} + \epsilon_1(l)$. The overall advantage of the simulator in D-Linear game is $\frac{1}{2} \times Pr[\mu = \mu'|\mu = 0] + \frac{1}{2} \times Pr[\mu = \mu'|\mu = 1]$ - $\frac{1}{2} = \frac{1}{2} \times \frac{1}{2} + \frac{1}{2} \times (\frac{1}{2} + \epsilon_1(l))$ - $\frac{1}{2} = \frac{\epsilon_1(l)}{2}$.

Therefore, if the $\mathcal{A}$ has a non-negligible advantage $\epsilon_1(l)$ in the above game then we can build a simulator (S) which can break the D-Linear problem with non-negligible quantity $\frac{\epsilon_1(l)}{2}$. Hence, the theorem. □

Theorem 2. *The proposed improved scheme provides receiver anonymity in IND-sCP game under the DDH assumption if there is no polynomial time adversary $\mathcal{A}$ who can distinguish a valid ciphertext and a random element with non-negligible advantage $Adv_{\mathcal{A}}(l)$ in security parameter l.*

Proof. In the second theorem we prove that the cipher components provides receiver anonymity. We prove in the following game that even if an attacker gains a valid decryption key, he is only able to find a correct match with C'. The attacker can not find out the underlying access policy. This proof strengthen our claim that even if any user is able to find himself as the intended recipient for a ciphertext, he is not able to find out who else are the other intended recipients for the same ciphertext. We show that the cipher components generated for an access policy are indistinguishable from an element chosen randomly from the group.

We first let the challenger set the groups G and G_T of prime order p, with an efficient bilinear map, e and generator g. The challenger flips a binary coin μ outside of $\mathcal{S}$ view and assigns a tuple $(g, A = g^a, B = g^b, Z)$ to $\mathcal{A}$. If $\mu = 1$ then the challenger sets Z as g^{ab} else a random value with equal probability.

Init: The simulator $\mathcal{S}$ runs $\mathcal{A}$. $\mathcal{A}$ commits two access policies W_0^* and W_1^* for which he wishes to be challenged upon.

In W_0^* and W_1^* for one attribute λ, $W_{0,\lambda}^* \neq W_{1,\lambda}^*$. There is at least one value $v_{\lambda,r}$ from value set of attribute λ, such that $v_{\lambda,r} \notin W_{0,\lambda}^*$ and $v_{\lambda,r} \in W_{1,\lambda}^*$. Here $1 \leq r \leq n_\lambda$. For rest of the attributes $W_{0,i}^* = W_{1,i}^*$ where $1 \leq i \leq n$ and $i \neq \lambda$.

Setup: $\mathcal{S}$ takes the following values: $g_1 = g$, $g_2 = g^{x_2}$. Here, x_2 is chosen randomly from $\mathbb{Z}_p$. $H(i \| v_{i,j})$ is computed as an output of an random oracle. Another two random secret values are chosen as α and β. The simulator $\mathcal{S}$ announces the public key as: $g_1 = g$, $g_2 = g^{x_2}$, $g_1^\alpha = g^\alpha$, $g_2^\beta = g^{x_2\beta}$, $\mathrm{e}(g_1,g_2)^\alpha = \mathrm{e}(g,g)^{x_2\alpha}$.

Preprocessing Phase: The attacker $\mathcal{A}$ collects following results in response of his queries made to simulator.

- Whenever $\mathcal{A}$ makes its k^{th} key generation query for the set L_k of attributes such that $F(L_k, W_0^*) = F(L_k, W_1^*)$. That is, $\mathcal{A}$ is allowed to issue a valid secret key for which the match procedure returns true with the challenge ciphertext components. However the restriction is imposed as the key should match with both the challenge access structure W_0^* and W_1^*. The simulator $\mathcal{S}$ selects a random value ρ and the key components are generated as follows.

$$D = (g_2 \prod_{i=1}^{n} H(i \| v_{i,j_i}))^\alpha \cdot g_2^\rho = (g^{x_2} \cdot g^{H'(i\|v_{i,j})})^\alpha \cdot g^{x_2\rho}$$

$$\bar{D} = g_1^{\frac{\rho}{\beta}} = g^{\frac{\rho}{\beta}}$$

- For the matching phase, $\mathcal{S}$ chooses random values s, and t from $\mathbb{Z}_p$. Then $\mathcal{S}$ provides the output of matching phase components for access policies W.
- The attacker issues the query for $H(i \| v_(i,j))$. For all the attribute values except $v_{\lambda,r}$, The simulator S runs the random oracle function and provides as output an element from G. The simulator S records the queries and its outputs. So that if any query is repeated by attacker then the same result as given for that query previously is repeated in output. For an query for $H(\lambda \| v_{\lambda,r})$ the output returned is $B = g^b$.

Challenge: Then $\mathcal{S}$ flips a random coin ν.

- $\mathcal{S}$ sets st as a and t is selected as any random value chosen from $\mathbb{Z}_p$. This results in $g_1^{st} = g^a = A$ and $g_2^{st\beta} = g^{x_2 a\beta} = A^{x_2\beta}$.
- The simulator $\mathcal{S}$ generates n shares of value t and use each share t_i for encrypting the values for attribute i. For all the cipher components $W_{\nu,i}^*$ where $1 \leq i \leq n$ and $i \neq \lambda$ the cipher components $\tilde{C_{i,j}}$ are generated as $g_2^{t_i} H(i \| v_{i,j})^{st} = g^{x_2 t_i} g^{H'(i\|v_{i,j})a}$.
- For values $v_{\lambda,j}$ of attribute $W_{\nu,\lambda}^*$ which are included in both $W_{0,\lambda}$ and $W_{1,\lambda}$, the cipher components $\tilde{C_{\lambda,j}}$ are generated as in the real scheme using the value of t_λ.
- For the value $v_{\lambda,r}$ which makes a differentiation between $W_{0,\lambda}$ and $W_{1,\lambda}$, the cipher component $\tilde{C_{\lambda,r}}$ is calculated as follows.
  - If $\nu = 0$ then $\tilde{C_{\lambda,r}}$ will be a random value. This is valid because $\tilde{C_{\lambda,r}}$ is not in $T_{0,\lambda}$ as per definition.
  - If $\nu = 1$ then $\tilde{C_{\lambda,r}}$ is set as $g^{x_2 t_\lambda} Z$. Here we have taken the output of $H(\lambda \| v(\lambda, j))$ from random oracle as g^b. If Z is a valid element with value g^{ab} then $\tilde{C_{\lambda,r}}$ will be a correct element else it will be a random element.
- C' is calculated as e$(g,g)^{ax_2\alpha}$.

The adversary will be given ciphertext components $\langle C', \hat{C}, \bar{C}, \{\{\tilde{C_{i,j}}\}_{1\leq j\leq n_i}\}_{1\leq i\leq n}\rangle$. At the end of the challenge phase the adversary uses following values $\langle\ g_1^{st} = g^a$, $H(\lambda\|v(\lambda,r))= g^b$, $C_{\lambda,r}\ \rangle$ to find out the underlying access policy. If $C_{\lambda,r}$ is a correct cipher component then it represents the value $g_2^{t_\lambda} \cdot g^{ab}$ else it is a random value.

Postprocessing Phase: $\mathcal{A}$ is allowed to run a number of queries for attribute keys and ciphertext components for the matching phase with the same conditions as imposed in the preprocessing phase.

Guess: $\mathcal{A}$ submits a guess ν' of ν. If $\nu' = \nu$, then $\mathcal{S}$ outputs μ=1 to indicate that it was given a valid DDH-tuple, else it outputs μ=0 to indicate that the ciphertext is a random element. Therefore, $\mathcal{A}$ gains no information about ν, in turn, $Pr[\nu \neq \nu' \mid \mu{=}0] = \frac{1}{2}$. As the simulator guesses μ'=0 when $\nu \neq \nu'$, $Pr[\mu = \mu' \mid \mu{=}0] = \frac{1}{2}$. If $\mu = 1$, then the adversary $\mathcal{A}$ is able to view a valid cipher components with advantage $\epsilon_2(l)$, a negligible quantity in security parameter in l. Therefore, $Pr[\nu = \nu'|\mu = 1] = \frac{1}{2} + \epsilon_2(l)$. Similarly, the simulator $\mathcal{S}$ guesses μ'=1 when $\nu = \nu'$, in turn, $Pr[\mu' = \mu|\mu = 1] = \frac{1}{2} + \epsilon_2(l)$. The overall advantage of the simulator in DDH game is $\frac{1}{2} \times Pr[\mu = \mu'|\mu = 0] + \frac{1}{2} \times Pr[\mu = \mu'|\mu = 1] - \frac{1}{2} = \frac{1}{2} \times \frac{1}{2} + \frac{1}{2} \times (\frac{1}{2} + \epsilon_2(l)) - \frac{1}{2} = \frac{\epsilon_2(l)}{2}$.

Therefore, if the $\mathcal{A}$ has a non-negligible advantage $\epsilon_2(l)$ in the above game then we can build a simulator (S) which can break the DDH problem with non-negligible quantity $\frac{\epsilon_2(l)}{2}$. Hence, the theorem. $\square$

7 Performance Analysis

We have experimented the proposed scheme on a Linux system with Intel core-i3 processor running at 2.30 GHz and 3 GB RAM. Pairings are constructed on the curve $y^2 = x^3 + x$ over the field F_q for some prime q = 3 mod 4. The order p of the groups G and G_T is a prime number 160 bits, while the length of q is 512 bits. The resultant time required in matching phase operation is around 0.04 to 0.08 s with respect to total number of attributes values ranging from 10 to 100.

The improved scheme facilitates a receiver to find out whether he is the intended recipient or not with just n multiplication operations and three bilinear pairing operations. Here, n denotes the number of attribute categories. The operation complexity of matching phase is $O(n) + O(1) \approx O(n)$. While comparing the improved scheme with Zhang *et al.*'s scheme we found the following results with respect to matching phase operation (Table 1).

Table 1. Comparison of the matching phase

Parameters (Used for match operation)	Zhang *et al.*'s scheme [1]	Proposed scheme
Number of user key components	$n+2$	2
Number of cipher components	$n_i \cdot n + 3$	$n_i \cdot n + 3$
Number of bilinear mapping operations	3	3
Number of multiplication operations	$2 \cdot n$	n

Here, n denotes the attribute categories in the system and n_i denotes the number of attribute values in i^{th} category ($1 \leq i \leq$ n).

We note that the performance comparison is done for the matching operation of the Zhang *et al.*'s scheme and the proposed improvement. The other AABE schemes [5–8] do not provide matching operation, as a result, the decryption procedure of these schemes are not efficient. The proposed improved scheme can be used with any existing AABE scheme to make the decryption procedure efficient.

8 Conclusion

We discussed anonymous attribute based encryption (AABE) schemes and found security flaws in a recently proposed AABE scheme [1]. The security weakness of the scheme [1] occurs in its matching phase that discloses the identity of the user. We proposed an improved scheme for the matching phase that keeps the scheme's anonymity feature intact. The proposed improved scheme can be incorporated in any AABE scheme in order to improve the efficiency of decryption procedure. User accountability is another important concern in AABE scheme, which can also be integrated in the proposed improvement and we left this as an interesting future scope of the proposed work.

References

1. Zhang, Y., Chen, X., Li, J., Wong, D.S., Li, H.: Anonymous attribute-based encryption supporting efficient decryption test. In: Proceedings of the ACM SIGSAC Symposium on Information, Computer and Communications Security, pp. 511–516 (2013)
2. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)
3. Bethencourt, J., Sahai, A., Waters, B.: Ciphertext-policy attribute-based encryption. In: Proceedings of IEEE Symposium on Security and Privacy, pp. 321–334 (2007)
4. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the ACM Conference on Computer and Communications Security, pp. 89–98 (2006)
5. Kapadia, A., Tsang, P.P., Smith, S.W.: Attribute-based publishing with hidden credentials and hidden policies. In: Proceedings of Network and Distributed System Security Symposium, pp. 179–192 (2007)
6. Yu, S., Ren, K., Lou, W.: Attribute-based content distribution with hidden policy. In: Proceedings of Workshop on Secure Network Protocols, pp. 39–44 (2008)
7. Nishide, T., Yoneyama, K., Ohta, K.: Attribute-based encryption with partially hidden encryptor-specified access structures. In: Bellovin, S.M., Gennaro, R., Keromytis, A.D., Yung, M. (eds.) ACNS 2008. LNCS, vol. 5037, pp. 111–129. Springer, Heidelberg (2008)
8. Li, J., Ren, K., Zhu, B., Wan, Z.: Privacy-aware attribute-based encryption with user accountability. In: Samarati, P., Yung, M., Martinelli, F., Ardagna, C.A. (eds.) ISC 2009. LNCS, vol. 5735, pp. 347–362. Springer, Heidelberg (2009)
9. Boneh, D., Boyen, X., Shacham, H.: Short group signatures. In: Franklin, M. (ed.) CRYPTO 2004. LNCS, vol. 3152, pp. 41–55. Springer, Heidelberg (2004)