User Collusion Avoidance Scheme for Privacy-Preserving Decentralized Key-Policy Attribute-Based Encryption

Yogachandran Rahulamathavan, Suresh Veluru, Jinguang Han, Fei Li, Muttukrishnan Rajarajan, and Rongxing Lu

***Abstract*—Decentralized attribute-based encryption (ABE) is a variant of multi-authority based ABE whereby any attribute authority (AA) can independently join and leave the system without collaborating with the existing AAs. In this paper, we propose a user collusion avoidance scheme which preserves the user's privacy when they interact with multiple authorities to obtain decryption credentials. The proposed scheme mitigates the well-known user collusion security vulnerability found in previous schemes. We show that our scheme relies on the standard complexity assumption (decisional bilienar Deffie-Hellman assumption). This is contrast to previous schemes which relies on non-standard assumption (q-decisional Diffie-Hellman inversion).**

***Index Terms*—Attribute-based encryption, user collusion.**

I. Introduction

Attribute-based encryption (ABE) supports both confidentiality and access control with a single encryption and has been identified as a potential technology for data sharing in distributed environment such as cloud [1], [2], [5]. Four parties namely data owner, users, attribute authority (AA) and storage server (i.e., cloud) involved in ABE. Users identities can be defined as a set of attributes (i.e., name, age, gender, location, passed driving test, purchased premium TV channels, etc). AA maintains encryption and decryption credentials for the users attributes. Data owner chooses a set of attributes based on the type of data and uses the corresponding encryption credentials for those attributes to encrypt the data. Then the encrypted data will be uploaded onto the cloud by the data owner. At the same time users obtain the decryption credentials for a set of attributes by proving to the AA that they are legitimate users for those attributes. If the users want to access the encrypted data, then they will first download the encrypted data from the cloud and will compare whether they satisfy the set of attributes defined by the data owner during the encryption.

The first ABE scheme was developed using single AA [2]. Later multiple AAs based ABE (MA-ABE) was proposed in [12], since single AA ABE scheme allowed too much information at single entity. In the MA-ABE scheme, there are multiple AAs responsible for disjoint sets of attributes. In conventional MA-ABE scheme, users interact with multiple AAs to obtain decryption credentials for their attributes. If there is no privacy protection for users then all AAs can share (collude) the particular user's information (attributes) to reveal the user's identity. Hence the focus of this paper is on achieving user privacy during the interaction with AAs (interesting readers can refer [1]–[5] for more details about ABE schemes). Let us provide concrete example applications for the proposed algorithm.

The first potential application is healthcare where ABE has been considered to provide fine-grain access control. The users of healthcare application could be staffs at hospital and patients i.e., they upload and retrieve sensitive data on daily basis [17]. Lets consider a scenario where diabetic patient wants to access his medical data which is in fact encrypted using ABE and one of the attributes is "Name of the Hospital". Unfortunately this hospital is known for diabetic clinic and obtaining a relevant key from AA (who may be a third party) responsible for "Name of the Hospital" attribute will jeopardize the patient's privacy straightaway. However, the proposed technique allow patients to access relevant information related to their condition without revealing their attributes (i.e., type of disease, hospital, gender, age) to the AAs.

The second potential application is smart grid where smart gridŠs control center can use ABE to broadcast a single encrypted message to a specific group of users (smart meters) [18]. Each user in the targeted group can individually decrypt the message based on their attributes. The attributes could be sensitive and reveal users day-to-day electricity usage e.g., attributes such as location, type of electrical equipment, subscription package can be passively leak user's private information. Because the same user might have different homes and businesses at different locations however some attributes such name, date of birth will be same for all properties. If that is the case then, with some side information and information resides at AAs, the user's spatial-temporal profile can be revealed by AAs. This vulnerability can be mitigated by the proposed scheme.

Within this context first known privacy-preserving (PP) decentralized ABE scheme was proposed in [6]. The scheme in [6] enables users to interact with AAs to obtain credentials for attributes via anonymous protocol. However, the solution in [6] is vulnerable to collusion attack [7] i.e., two users (colluding users) who have obtained credential for an attribute o from an AA O can generate credential for a third user for o without involving O. This is a serious security issue since that third user in fact may not satisfy o as part of his identity. Hence in this paper, we propose a scheme which mitigates the above user collusion attack using anonymous key issuing protocol. The main achievement of this work is on tieing secret known for AA and secret known for user in a non-linear fashion. This non-linear coupling withstands the colluding users to generate new keys for unauthorized users via algebraic operations.

Related Works: There are two main types of ABE namely ciphertext-policy ABE (CP-ABE) and key-policy ABE (KP-ABE) [2], [5] and the focus of this paper is on KP-ABE. The KP-ABE scheme can be further divided into two: single AA ABE and MA-ABE. Chase et. al. presented a MA-ABE scheme [11] using trusted central AA for distributing all the keys [11]. MA-ABE scheme without central AA has been presented by Chase and Chow in [12] where, each pair of AAs securely exchange a shared secret among them during the set up process. In [11], users

This work was supported by the Engineering and Physical Sciences Research Council of the U.K. under EPSRC Grant EP/K03345X/1 and National Natural Science Foundation of China under Grant 61300213.

Y. Rahulamathavan, F. Li, and M. Rajarajan are with the School of Engineering and Mathematical Science, City University London, London, U.K. (e-mail: {Yogachandran.Rahulamathavan.1, fei.li.1, r.muttukrishnan}@city.ac.uk).

S. Veluru is with the United Technologies Research Centre, Ireland. (e-mail: VeluruS@utrc.utc.com). Part of this work done while the author was with City University London.

J. Han is with the Jiangsu Provincial Key Laboratory of E-Business, Nanjing University of Finance and Economics, Nanjing, Jiangsu 210003, China (e-mail: jghan22@gmail.com).

R. Lu is with the Division of Communication Engineering, School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore (e-mail: rxlu@ntu.edu.sg).

must submit their global identities (GIDs) to each AA to obtain the decryption credentials. This will breach the user privacy since a set of corrupted AAs can pool together all the attributes belong to the particular GID.

In order to mitigate this privacy vulnerability, Chase and Chow proposed an anonymous key issuing protocol in [12], whereby a user can obtain the decryption keys from AAs without revealing her GID. Even though the scheme proposed by Chase and Chow eliminates the central AA, all the AAs must be online and collaborate with each other to set up the ABE system hence it is not fully decentralized. Similarly, various protocols have been proposed to decentralize the ABE scheme [9]–[12], however, each scheme has its own merits and demerits.

The first known fully decentralized (i.e., without central AA) MA-ABE scheme was proposed in [9]. In [9], any party can become an AA and there is no requirement for any global coordination other than the creation of an initial set of common reference parameters. Crucially [9] overcome the collusion vulnerability without any coordination between AAs by proposing novel techniques to tie key components together and prevent collusion attacks between users with different global identifiers. However, the scheme in [9] does not preserve the user privacy i.e., attributes of users can be collected by AAs by tracing users' GIDs. The scheme in [12] considers privacy but not fully decentralized. For the first time, Han et al. proposed a PP decentralized scheme for KP-ABE in [6]. In contrast to the existing decentralized ABE schemes, the scheme in [6] preserves the user privacy and relies only on decisional bilinear Diffie-Hellman (DBDH) standard complexity assumption.

In [6], the GID of the user is used to tie all the decryption keys together, where blind key generation protocol has been used to issue the decryption keys. Hence, corrupted AAs cannot pool the users' attributes by tracing the GIDs' of the users from the decryption keys. Unfortunately, the scheme in [6] cannot prevent user collusion, hence, two users can pool their decryption keys to generate decryption keys for an unauthorized user [7]. This is due to weak bind between users' GID and the decryption keys.

In this paper, contrasting to all the works in literature, we propose a PP decentralized KP-ABE scheme in order to mitigate the known user collusion security vulnerability [7]. We exploit the anonymous key issuing protocol in [12] to strengthen the bind between decryption keys and GID as well as to preserve the user privacy. In order to incorporate the anonymous key issuing protocol, we modify the PP decentralized KP-ABE scheme in [6]. We prove this by contradiction that the proposed scheme is secure i.e., we reduce the DBDH standard complexity assumption to show that an adversary who can break the proposed scheme can be exploited to break the DBDH assumption. We also proved that the anonymous key issuing protocol is free from leak, selective-failure and avoid user collusion.

II. Decentralized Key-Policy Attribute-based Encryption

The following notations are used throughout this paper: we use $x \xleftarrow{R} X$ to denote that x is randomly selected from X. Suppose $\mathbb{Z}_p$ is a finite field with prime order p, by $\mathbb{Z}_p[x]$, we denote the polynomial ring on $\mathbb{Z}_p$. We used the following well-known techniques as building blocks for our scheme: Lagrange Interpolation, bilinear groups, decisional bilinear Diffie-Hellman complexity assumption, commitment scheme, proof-of-knowledge and access tree (refer [2], [6] for more details).

In a decentralized scheme, any AA can join and/or leave the system at any time without rebooting the system. First of all, we will explain sub-algorithms and security game, original scheme in [6] followed by our PP decentralized KP-ABE. In Section III, we incorporate the anonymous key issuing protocol to strengthen the bind between user GID and decryption keys.

Global Setup $(\mathcal{GS})-$ For a given security parameter λ, $\mathcal{GS}$ generates the bilinear groups $\mathbb{G}_1$ and $\mathbb{G}_2$ with prime order p as follows: $\{\mathbb{G}_1, \mathbb{G}_2\} \leftarrow \mathcal{GS}(1^\lambda)$. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$ be a bilinear map and g, h and h_1 be the generators of $\mathbb{G}_1$. There are N number of authorities $\{A_1, \ldots, A_N\}$: A_k monitors n_k attributes i.e. $\tilde{A}_k = \{a_{k,1} \ldots, a_{k,n_k}\}, \forall k$.

AAs Setup $(\mathcal{AS})-$ Security parameters of A_k: $SK_k = \left\{\alpha_k, \beta_k, \text{ and } [t_{k,1}, \ldots, t_{k,n_k}]\right\} \xleftarrow{R} \mathbb{Z}_p, \forall k$. Public parameters of A_k: $PK_k = \{Y_k = e(g,g)^{\alpha_k}$, $Z_k = g^{\beta_k}$, and $[T_{k,1} = g^{t_{k,1}}, \ldots, T_{k,n_k} = g^{t_{k,n_k}}]\}$, $\forall$ k. Each A_k specifies an (k_k, n_k) threshold access structure where $k_k \leq n_k$.

Key Generation $(\mathcal{KG})-$ Collision-Resistant Hash Function $H : \{0,1\}^* \rightarrow \mathbb{Z}_p$ to generate u from the user global identity. Attribute set of user is $\tilde{A}_u$: $\tilde{A}_u \cap \tilde{A}_k = \tilde{A}_u^k \ \forall \ k$. A_k generates $r_{k,u} \in_R \mathbb{Z}_p$ and $d_k = k_k - 1$ degree polynomial q_k where k_k – threshold value of that node. with $q_k(0) = r_{k,u}$. Now decryption keys for the user u is generated as follows: $D = D_{k,u} = g^{\alpha_k} h^{r_{k,u}} h_1^{u\beta_k}, D_{k,j} = h^{\frac{q_k(a_{k,j})}{t_{k,j}}}, \forall a_{k,j} \in \tilde{A}_u^k$.

Encryption $(\mathcal{E})-$ Attribute set for the message m is $\tilde{A}_m$: $\tilde{A}_m \cap \tilde{A}_k = \tilde{A}_m^k, \forall k$, i.e. $\tilde{A}_m = \{\tilde{A}_m^1, \ldots, \tilde{A}_m^k, \ldots, \tilde{A}_m^N\}$.Data owner of message m randomly chooses $s \in_R \mathbb{Z}_p$, and output the ciphertext as follows: $C = C_1 = m. \prod_{k \in I_C} e(g,g)^{\alpha_k s}, C_2 = g^s, C_3 = \prod_{k \in I_C} g^{\beta_k s}$and $\left\{C_{k,j} = T_{k,j}^s\right\}_{\forall k \in I_C, a_{k,j} \in \tilde{A}_m^j}$ where I_c denotes the index set of the authorities.

Decryption $(\mathcal{D})-$ In order to decrypt C, the user u, computes X, Y and S_k as follows: $X = \prod_{k \in I_C} e\left(C_2, D_{k,u}\right)$, $Y = e\left(C_3, h_1^u\right)$ and $S_k = \prod_{a_{k,j} \in A_m^k} e\left(C_{k,j}, D_{k,j}\right)^{\Delta_{a_{k,j}, \tilde{A}_m^j}(0)}$. User then decrypts the message m as follows: $m = \frac{C_1 Y \prod_{k \in I_C} S_k}{X}$.

Fig. 1. The original decentralized key-policy attribute-based encryption scheme [6].

A. Sub-algorithms

In general decentralized schemes contain the following five sub-algorithms:

Global Setup: This algorithm takes a security parameter as input and output system parameters. These system parameters can be used by AAs who join the system.

AA Setup: Each AA uses the system parameters obtained from the global setup to generate public and private keys for the attributes it maintains.

Key Issuing: User and AA interact via anonymous key issuing protocol (see Section III) in order to determine attributes belong to the user. Then AA generates decryption credentials for those attributes and send them to the user.

Encryption: The encryption algorithm takes a set of attributes maintained by AA and the data as input. Then it outputs the ciphertext of the data.

Decryption: The decryption algorithm takes the decryption credentials received from AAs and the ciphertext as input. The decryption will be successful if and only if the user attributes satisfy the access structure.

B. Security Game

In order to avoid the security vulnerabilities, ABE schemes should be proven to be secure against the selective identity (ID) model [1]. In the selective ID model, the adversary should provide the identities of the AAs (challenge identities) he wishes to challenge the challenger with. Then challenger (i.e., the system) will generate necessary parameters corresponding to the challenge identities and send them to the adversary. Then the adversary is allowed to make secret queries about the challenge identities. If the adversary cannot decrypt the encrypted message at the end with non-negligible advantage then the proposed scheme is secure against the selective ID model. Formally, this is represented by the following game between the adversary and the challenger:

Setup- Adversary sends a list of attribute sets and AAs including corrupted AAs to the challenger. Now the challenger generates public and private keys corresponding to the attributes and AAs provided by the adversary. Challenger provides public and private keys corresponding to the corrupted AAs to the adversary while only public keys corresponding to the remaining AAs to the adversary.

Secret Key Queries- The adversary is allowed to make any number of secret key queries as he wants against the AAs. However, the only requirement is that for each user, there must be at least one non corrupted attribute AA from which the adversary can get insufficient number of secret keys.

Challenge- The adversary sends two messages m_0 and m_1 to the challenger in plain domain. Now the challenger randomly chooses one of the messages and encrypt it and send the ciphertext to the adversary.

More Secret Key Queries- The adversary is allowed to make more secret key queries as long as he satisfies the requirement given earlier.

Guess- Now the adversary guesses which message was encrypted by the challenger. The adversary is said to be successful if he guesses the correct message with probability $\frac{1}{2} + \epsilon$ whereby ϵ is a non-negligible function.

C. Construction of our new algorithm

Let us consider N number of AAs and denote them as $A_1, \ldots, A_N$. The attribute set managed by the AA A_k is denoted as $\widetilde{A_k} = \{a_{k,1}, \ldots, a_{k,n_k}\}$ $\forall k$. Let us first describe the original algorithm in [6] in Fig. 1. Then in Fig. 2 we show how our algorithm differ from [6]. Let us explain the important steps involved in Fig. 1 and how our algorithm in Fig. 2 differs from [6].

Initially, for a given security parameter λ, global setup algorithm ($\mathcal{GS}$) generates the bilinear groups $\mathbb{G}_1$ and $\mathbb{G}_2$ with prime order p i.e., $\{\mathbb{G}_1, \mathbb{G}_2\} \leftarrow \mathcal{GS}(1^\lambda)$. The AA setup algorithm ($\mathcal{AS}$) is executed by each AA to randomly generate public keys (PK) and the corresponding secret keys (SK). The public-secret key pairs for A_k is given as $\{(Y_k, Z_k, [T_{k,1}, \ldots, T_{k,n_k}]), (\alpha_k, \beta_k, [t_{k,1}, \ldots, t_{k,n_k}])\}$.

Let us denote the attribute set belongs to user u as $\widetilde{A_u}$ and the common attribute set between user u and AA k as $\widetilde{A_u^k}$ i.e., $\widetilde{A_u^k} = \widetilde{A_u} \bigcap \widetilde{A_k}$. Key generation ($\mathcal{KG}$) algorithm will be used to issue decryption keys to the user u with a set of attributes $\widetilde{A_u}$. The algorithm outputs a key that enables the user to decrypt a message encrypted under a set of attributes $\widetilde{A_u^k}$ based on threshold policy i.e., if users satisfy more attributes than AA's defined threshold then the user can successfully generate decryption credential. Since threshold policy is very basic, we use access tree based access structure in our scheme.

Let us briefly explain the access tree. First choose a polynomial q_x for each node x (including the leaves) in the tree $\mathbb{T}$. These polynomials are chosen in the following way in a top-down manner, starting from the root node r. For each node x in the tree, set the degree d_x of the polynomial q_x to be one less than the threshold value k_x of that node, that is, $d_x = k_x - 1$. Now, for the root node r, set $q_r(0) = r_{k,u}$ and d_r other points of the polynomial q_r randomly to define it completely. For any other node x, set $q_x(0) = q_{parent(x)}(index(x))$ and choose d_x other points randomly to completely define q_x. Once the polynomials have been decided, for each leaf node x, we give the secret value $D_{k,u}$, $D^1_{k,u}$ and $D^j_{k,u}$, $\forall a_{k,j} \in \widetilde{A_u^k}$ as shown in Fig. 2 to the user. Hence user can decryption keys if and only if $\mathbb{T}(\widetilde{A_u^k}) = 1$.

Global Setup ($\mathcal{GS}$)$-$ Same as the original.
AAs Setup ($\mathcal{AS}$)$-$ The initial setup is same as the original. We consider more generic tree access structure. Hence we replace the threshold access structure requirement into tree based as follows: A_k specify m_k as minimum number of attributes required to satisfy the access structure ($m_k \leq n_k$).

Key Generation ($\mathcal{KG}$)$-$ Attribute set of user is $\tilde{A}_u$: $\tilde{A}_u \cap \tilde{A}_k = \tilde{A}^k_u$ $\forall$ k. A_k generates $r_{k,u} \in_R \mathbb{Z}_p$ and polynomial q_x for each node x (including the leaves) $\mathbb{T}$. For each node x, the degree d_x of the polynomial q_x is $d_x = k_x - 1$ where k_x – threshold value of that node. Now, for the root node r, set $q_r(0) = r_{k,u}$. For any other node x, set $q_x(0) = q_{parent(x)}(index(x))$. Now decryption keys for the user u is generated as follows: $D = D_{k,u} = g^{-\alpha_k} h^{\frac{\beta_k}{r_{k,u}+u}} h_1^{\frac{r_{k,u}}{\beta_k+u}}, D^1_{k,u} = h^{\frac{1}{r_{k,u}+u}}, D^j_{k,u} = h_1^{\frac{q_{a_{k,j}}(0)}{(\beta_k+u)t_{k,j}}}, \forall a_{k,j} \in \tilde{A}^k_u$.

Encryption ($\mathcal{E}$)$-$ This exactly same as the original scheme.
Decryption ($\mathcal{D}$)$-$ In order to decrypt C, the user u, computes X, Y and S_k as follows: $X = \prod_{k \in I_C} e\left(C_2, D_{k,u}\right)$, $Y = e\left(C_3, D^1_{k,u}\right)$ and $S_k = \prod_{a_{k,j} \in A^k_m} e\left(C_{k,j}, D^j_{k,u}\right)^{\Delta_{a_{k,j}, \tilde{A}^j_m}(0)}$.
User then decrypts the message m as follows: $m = \frac{C_1 X}{Y \prod_{k \in I_C} S_k}$.

Fig. 2. The proposed decentralized KP-ABE scheme. This shows the difference between the original and the proposed algorithm.

Let us denote the set of attributes used to encrypt message m as $\widetilde{A_m}$ and the common attribute set between message m and the AA k as $\widetilde{A_m^k}$ i.e., $\widetilde{A_m} = \{\widetilde{A_m^1}, \ldots, \widetilde{A_m^k}, \ldots, \widetilde{A_m^N}\}$. Let us also denote the index set of AAs involved in the ciphertext of message m as I_c. The encryption algorithm ($\mathcal{E}$) in Fig. 2 encrypts the message $m \in \mathbb{G}_2$ using an attribute set $\widetilde{A_m}$. In order to encrypt the message, the message owner randomly generates s and computes ciphertext $C = \left[C_1, C_2, C_3, C_{k,j}, \forall a_{k,j} \in \widetilde{A_m^k}\right]$. If the user has decryption keys for the attributes of message m then he can obtain the message m from the ciphertext using the following four steps by executing the decryption algorithm ($\mathcal{D}$). First, user can use decryption key $D_{k,u}$ and C_2 to compute X as

$$X = \prod_{k \in I_C} e\left(C_2, D_{k,u}\right) = \prod_{k \in I_C} e\left(g^s, g^{-\alpha_k} h^{\frac{\beta_k}{r_{k,u}+u}} h_1^{\frac{r_{k,u}}{\beta_k+u}}\right),$$
$$= \prod_{k \in I_C} e\left(g, g\right)^{-s\alpha_k} \prod_{k \in I_C} e\left(g, h\right)^{\frac{s\beta_k}{r_{k,u}+u}} \prod_{k \in I_C} e\left(g, h_1\right)^{\frac{sr_{k,u}}{\beta_k+u}},$$

then user uses decryption key $D^1_{k,u}$ and C_3 to compute Y as

$$Y = e\left(C_3, D^1_{k,u}\right) = e\left(\prod_{k \in I_C} g^{\beta_k s}, h^{\frac{1}{r_{k,u}+u}}\right) = \prod_{k \in I_C} e\left(g, h\right)^{\frac{s\beta_k}{r_{k,u}+u}},$$

and uses $D^{j}_{k,u}$, $C_{k,j}$, $\forall a_{k,j} \in \widetilde{A^{j}_{m}}$ and polynomial interpolation to get $r_{k,u}$ as

$$\begin{aligned} S_k &= \prod_{a_{k,j} \in \widetilde{A^{k}_{m}}} e\left(C_{k,j}, D^{j}_{k,u}\right)^{\Delta_{a_{k,j} \in \widetilde{A_m}, \widetilde{A^{j}_{m}}}(0)}, \\ &= \prod_{a_{k,j} \in \widetilde{A^{k}_{m}}} e(g, h_1)^{\frac{s}{(\beta_k + u)} q_{a_{k,j}}(0) \Delta_{a_{k,j} \in \widetilde{A_m}, \widetilde{A^{j}_{m}}}(0)}, \end{aligned}$$

where $q_{a_{k,j}}(0) = q_{parent(a_{k,j})}(index(a_{k,j}))$. Hence, $S_k = e(g, h_1)^{\frac{s r_{k,u}}{(\beta_k + u)}}$. Now user can get the message m using C_1 and pre-computed values $X,\ Y,\ S_k,\ \forall k$ as follows:

$$\frac{C_1 X}{Y \prod\limits_{k \in I_C} S_k} = \left(\frac{m. \prod\limits_{k \in I_C} e(g,g)^{\alpha_k s}}{\prod\limits_{k \in I_C} e(g,h)^{\frac{s\beta_{k,u}}{r_k + u}}} \right) \times$$

$$\left(\frac{\prod\limits_{k \in I_C} e\left(g,g\right)^{-s\alpha_k} \prod\limits_{k \in I_C} e\left(g,h\right)^{\frac{s\beta_k}{r_{k,u}+u}} \prod\limits_{k \in I_C} e\left(g,h_1\right)^{\frac{s r_{k,u}}{\beta_k + u}}}{\prod\limits_{k \in I_C} e(g,h_1)^{\frac{s r_{k,u}}{\beta_k + u}}} \right) = m.$$

D. Security Analysis

Theorem 1. *The proposed scheme is semantically secure against chosen plain text attack (CPA) in the selective ID model, if there exist negligible function υ such that, any adversary will succeed the security game explained earlier with probability at most $\frac{1}{2} + \upsilon$.*

Proof. Suppose if there is a probabilistic polynomial time adversary who can break our algorithm then there will be a challenger who can break the DBDH assumption by exploiting the adversary. Lets assume that the challenger is provided with $[g^a, g^b, g^c, Z]$ and if the challenger wants to break the DBDH assumption then he needs to determine whether $Z = \hat{e}(g,g)^{abc}$ or not with at least $\frac{1}{2} + \upsilon$ probability.

Let us assume that there is an adversary who can break the proposed algorithm. In this section, we will show that the challenger can use such an adversary to break the DBDH assumption. In order to exploit such an adversary, the challenger needs to incorporate the given $[g^a, g^b, g^c, Z]$ within the proposed algorithm (i.e., Fig. 2). First of all, let us explain how the challenger incorporates $[g^a, g^b, g^c, Z]$ within the global setup, AA setup, and key generation sub-algorithms. We stress here that this incorporation is indistinguishable from the steps provided in Fig. 2.

Initially, as explained in the security game, the adversary must submit a set of attributes and a set of AAs he wants to challenge. Let us denote the set of attributes provided by the adversary as $\widetilde{A_C} = \{\widetilde{A^1_C}, \ldots, \widetilde{A^k_C}, \ldots, \widetilde{A^N_C}\}$ where $\widetilde{A^k_C} = \widetilde{A_C} \bigcap \widetilde{A_k}$ and a list of corrupted AAs as C_A. One of the conditions as given in security game is that at least there will be one honest AA for each user whereby the adversary can get insufficient number of decryption credentials [11]. Let us denote GID of a particular user as ω and the corresponding honest AA as κ, and it's access structure $\mathbb{T}^{\kappa}$ hence $\mathbb{T}^{\kappa}(\widetilde{A^{\kappa}_C} \cap \widetilde{A^{\kappa}_{\omega}}) = 0$. Hence, we can divide the AAs who maintain attributes A_C into three: corrupted AAs, AAs who are not corrupted and not κ, and κ.

Firstly, challenger generates two random values $\gamma, \eta \xleftarrow{R} \mathbb{Z}_p$ and sets $h = g^a g^{\gamma}$ and $h_1 = g^{a^{\eta}} = g^{a\eta}$. For the corrupted AAs $A_k \in C_A$: The challenger generates $\upsilon_k, \beta_k, w_{k,j} \xleftarrow{R} \mathbb{Z}_p$ and sets $Y_k = e(g,g)^{\upsilon_k}$, $Z_k = g^{\beta_k}$ and $\{T_{k,j} = g^{w_{k,j}}\}_{a_{k,j} \in \widetilde{A_k}}$. Hence, the public-secret key pairs for $A_k \in C_A$ is given as $\{(\upsilon_k, \beta_k, [w_{k,1}, \ldots, w_{k,n_k}]), (Y_k, Z_k, [T_{k,1}, \ldots, T_{k,n_k}])\}$. Challenger provides the secret-public key pairs of the corrupted AAs to the adversary. Hence, the adversary can compute $D_{k,\omega}, D^1_{k,\omega}$, and $D^j_{k,\omega}, \forall a_{k,j} \in \widetilde{A^k_{\omega}}$ himself for user ω without interacting with the challenger.

For the AAs who are not corrupted and not κ (i.e., $A_k \notin C_A \cup \kappa$): Challenger generates $\upsilon_k, \beta_k, w_{k,j} \xleftarrow{R} \mathbb{Z}_p$ and sets $Y_k = e(\underset{\sim}{g}^b, \underset{\sim}{g^{\upsilon_k}}) = e(g,g)^{b\upsilon_k}$, and $Z_k = g^{\beta_k}$. For the attributes $a_{k,j} \in \widetilde{A_C} \cap \widetilde{A_k}$, the challenger sets $T_{k,j} = g^{w_{k,j}}$. Other attributes i.e., $a_{k,j} \in \widetilde{A_k} - \widetilde{A^k_C}$, the challenger sets $T_{k,j} = h_1^{w_{k,j}} = g^{a\eta w_{k,j}}$. Hence, the public-secret key pairs for $A_k \notin C_A \cup \kappa$ is given as $\{(b\upsilon_k, \beta_k, w_{k,1}, \ldots, w_{k,n_k}), (Y_k, Z_k, T_{k,1}, \ldots, T_{k,n_k})\}$. Challenger sends public keys $(Y_k, Z_k, T_{k,1}, [T_{k,1}, \ldots, T_{k,n_k}])$ to the adversary. Now AA A_k assigns a polynomial q_x with degree d_x for every node in it's access tree $\mathbb{T}^k$. It first sets up a polynomial q_x of degree d_x for the root node x. It sets $q_x(0) = r_{k,\omega}$ and then sets rest of the points randomly to completely fix q_x. Now it sets polynomials for each child node x' of x as follows: $q_{x'}(0) = q_x(index(x'))$. Then computes $D_{k,\omega} = g^{-b\upsilon_k} g^{\frac{(a+\gamma)\beta_k}{r_{k,\omega}+\omega}} g^{\frac{a\eta r_{k,\omega}}{\beta_k+\omega}}, D^1_{k,\omega} = g^{\frac{a+\gamma}{r_{k,\omega}+\omega}}$, and $D^j_{k,\omega} = g^{\frac{a\eta q_{parent(a_{k,j})}(index(a_{k,j}))}{(\beta_k+\omega)w_{k,j}}}, \forall a_{k,j} \in \widetilde{A_C} \cap \widetilde{A_k}$ or $D^j_{k,\omega} = g^{\frac{q_{parent(a_{k,j})}(index(a_{k,j})))}{(\beta_k+\omega)w_{k,j}}}, \forall a_{k,j} \in \widetilde{A_k} - \widetilde{A^k_C}$

If $A_k = \kappa$: Challenger generates $\beta_{\kappa}, w_{\kappa,j} \xleftarrow{R} \mathbb{Z}_p$. Challenger sets $Y_{\kappa} = e(g^a, g^b) \prod\limits_{j=1, j\neq\kappa}^{n} Y_j^{-1} = e(g,g)^{\left(ab - \sum\limits_{A_k \notin C_A \cup \kappa} b\upsilon_k - \sum\limits_{A_k \in CA} \upsilon_k\right)}$, $Z_{\kappa} = g^{\beta_{\kappa}}$. For the attributes $a_{\kappa,j} \in \widetilde{A_C} \cap \widetilde{A_{\kappa}}$, the challenger sets $T_{\kappa,j} = g^{w_{\kappa,j}}$. Other attributes i.e., $a_{\kappa,j} \in \widetilde{A_{\kappa}} - \widetilde{A^{\kappa}_C}$, the challenger sets $T_{\kappa,j} = h_1^{w_{\kappa,j}} = g^{(a\eta)w_{\kappa,j}}$. Hence, the public-secret key pairs for $A_k = \kappa$ is given as $[(ab - \sum\limits_{A_k \notin C_A \cup \kappa} b\upsilon_k - \sum\limits_{A_k \in CA} \upsilon_k), \beta_k, w_{k,1}, ..., w_{k,n_k}, (Y_{\kappa}, Z_{\kappa}, T_{\kappa,1}, ...)]$. Challenger sends public keys $(Y_{\kappa}, Z_{\kappa}, T_{\kappa,1}, ...)$ to the adversary.

Since $\mathbb{T}^{\kappa}(\widetilde{A^{\kappa}_C} \cap \widetilde{A^{\kappa}_{\omega}}) = 0$, it should be stressed that maximum $d_x - 1$ number of children of node x could be satisfied. It first defines a polynomial q_x of degree d_x for the root node x such that $q_x(0) = r'$. For each satisfied child x' of x, the procedure chooses a random point $e_{x'}$ and sets $q_x(index(x')) = e_{x'}$. We will show later in this section that it is possible to implicitly define remaining children nodes using polynomial interpolation. Now it recursively defines polynomials for the rest of the nodes in the tree as follows. For each child node x' of x, $q_{x'}(0) = q_x(index(x'))$. Computes $D_{\kappa,\omega} = g^{\left(-ab + \sum\limits_{A_k \notin C_A \cup k*} b\upsilon_k + \sum\limits_{A_k \in CA} \upsilon_k\right)} g^{\frac{(a+\gamma)\beta_{\kappa}}{(r+\omega)}} g^{\frac{a\eta r'}{\beta_{\kappa}+\omega}}, D^1_{\kappa,\omega} = g^{\frac{(a+\gamma)}{(r+\omega)}}$, and $D^j_{\kappa,\omega} = h_1^{\frac{q_{parenta_{\kappa,j}}(index(a_{\kappa,j}))}{w_{\kappa,j}}} \forall a_{\kappa,j} \in A_C \cap \widetilde{A_k}$ or $D^j_{\kappa,\omega} = g^{\frac{r'\Delta_{0,S}(a_{\kappa,j}) + \sum_{i=1}^{d_{\kappa}-1} e_i \Delta_{i,S}(a_{\kappa,j})}{w_{\kappa,j}}}, \forall a_{\kappa,j} \in \widetilde{A_k} - \widetilde{A^k_C}$ where $r' = r + b\frac{(\beta_{\kappa}+\omega)}{\eta}$.

The following lemma proves $D_{\kappa,\omega}$ and $D^j_{\kappa,\omega}$ are correct and the challenger can generate these without knowing a, b, c from g^a, g^b, g^c.

Lemma 1. *$D_{\kappa,\omega}$ and $D^j_{\kappa,\omega}$ are correct and they can be generated by challenger without knowing a, b, c from g^a, g^b, g^c.*

Proof. In the beginning of the access structure we chosen e_1, e_2, $\ldots$, $e_{d_{\kappa}-1}$ random values for children nodes of root node and we set $q_x(0) = r' = r + b\frac{(\beta_{\kappa}+\omega)}{\eta}$. Hence, using the polynomial interpolation technique, we can implicitly assign values to other children nodes of root using the following valid polynomial function:

$q_x = r'\Delta_{0,S}(x) + \sum_{i=1}^{d_\kappa-1} e_i\Delta_{i,S}(x), S \in \{0,1,2,\dots,d_\kappa-1\}$. During the key extraction, for the correctness, the challenger hides r' in attribute $a_{\kappa,j} \in \{\widetilde{A_k} - \widetilde{A^k_C}\}$. Since, $T_{\kappa,j} = h_1^{w_{\kappa,j}}$, the $D^j_{\kappa,\omega}$ $\forall\ a_{\kappa,j} \in \{\tilde{A}_k - \tilde{A}^k_C\}$ is given as

$$D^j_{\kappa,\omega} = g^{\frac{q_{parent(a_{\kappa,j})}(index(a_{\kappa,j}))}{w_{\kappa,j}}} = g^{\frac{r'\Delta_{0,S}(a_{\kappa,j}) + \sum_{i=1}^{d_\kappa-1} e_i\Delta_{i,S}(a_{\kappa,j})}{w_{\kappa,j}}},$$

which is valid and identical to that in the original scheme. Now we will prove that the challenger can generate $D^j_{\kappa,\omega}$ and $D_{\kappa,\omega}$ for $a_{\kappa,j} \in \{\widetilde{A_k} - \widetilde{A^k_C}\}$ without knowing $a, b,$ and c.

$$\begin{aligned} D^j_{\kappa,\omega} &= g^{\frac{\left(r+b\frac{(\beta_\kappa+\omega)}{\eta}\right)\Delta_{0,S}(a_{\kappa,j}) + \sum_{i=1}^{d_\kappa-1} e_i\Delta_{i,S}(a_{\kappa,j})}{w_{\kappa,j}}}, \\ &= \left[g^r\left(g^b\right)^{\left(\frac{\beta_\kappa+\omega}{\eta}\right)}\right]^{\frac{\Delta_{0,S}(a_{\kappa,j})}{w_{\kappa,j}}} \prod_{i=1}^{d_\kappa-1} g^{\frac{e_i\Delta_{i,S}(a_{\kappa,j})}{w_{\kappa,j}}}, \end{aligned}$$

and

$$\begin{aligned} D_{\kappa,\omega} &= g^{\left(-ab + \sum_{A_k \notin C_A \cup \kappa} bv_k + \sum_{A_k \in CA} v_k\right)} g^{\frac{(a+\gamma)\beta_\kappa}{(r+\omega)}} g^{\frac{a\eta r'}{\beta_\kappa+\omega}} \\ &= g^{\left(-ab + \sum_{A_k \notin C_A \cup \kappa} bv_k + \sum_{A_k \in CA} v_k\right)} g^{\frac{(a+\gamma)\beta_\kappa}{(r+\omega)}} g^{\frac{a\eta(r+b\frac{(\beta_\kappa+\omega)}{\eta})}{\beta_\kappa+\omega}} \\ &= \left(g^b\right)^{\left(\sum_{A_k \notin C_A \cup \kappa} v_k\right)} g^{\left(\sum_{A_k \in CA} v_k\right)} h^{\frac{\beta_\kappa}{r+\omega}} h_1^{\frac{r}{\beta_\kappa+\omega}} \qquad \blacksquare \end{aligned}$$

Once adversary received all the credential from the challenger, he will send two messages, m_0 and m_1 to the challenger. Now the challenger randomly chooses one of the messages and encrypts it and sends the encrypted message back to the adversary. Let us denote the message chosen by the challenger as m_b where $b = 0$ or 1 and the encrypted message as $C = \{C_1 = m_b.Z, C_2 = g^c, C_3 = \prod_{k \in A_C} (g^c)^{\beta_k}$ and $\{C_{k,j} = (g^c)^{w_{k,j}}\}_{\forall k \in I_C, a_{k,j} \in \tilde{A}^j_m}\}$. We stress here that C is a valid encryption of the message m_b if $Z = e(g,g)^{abc}$. Hence, the adversary should have his usual non-negligible advantage υ of correctly identifying the message m_b. However, when $Z \neq e(g,g)^{abc}$, then C is just random value, hence, the adversary can have no more than $\frac{1}{2}$ probability of guessing correctly. Hence, if the adversary guesses correctly then challenger guesses that $Z = e(g,g)^{abc}$ and if adversary is wrong then challenger guesses that $Z \neq e(g,g)^{abc}$, hence, the challenger has an advantage of $\frac{\upsilon}{2}$ in distinguishing whether $Z = e(g,g)^{abc}$. Hence, an adversary who breaks our scheme with advantage υ implies an algorithm for breaking DBDH assumption with non-negligible advantage $\frac{\upsilon}{2}$. We can conclude that the proposed scheme is selective ID secure.

III. Anonymous Key Issuing Protocol

In order to issue keys to the users, Han et. al. [6] used key extraction protocol. The key extraction protocol joins secrets known to the user and AA via simple operation which let the users to collude successfully. To avoid the user collusion, we have to use schemes which support non-linear coupling of secrets, hence we are considering anonymous key issuing protocol based on anonymous credential system (See Section III-C for more details). In an anonymous credential system [14], the users can obtain and prove the possession of credentials while remaining anonymous. In such work it is assumed that each user has a unique secret key i.e., GIDs. Then the user can interact with each AA under a different pseudonym (generated using GID) in such a manner that it is impossible to link multiple pseudonyms belonging to the same user. In the proposed work, we exploited the anonymous credentials to allow users to obtain decryption keys from the AAs without revealing their GIDs.

In particular, in the proposed key issuing protocol, we coupled the properties of anonymous credential system with ABE. Hence, the user can obtain a set of decryption keys for his secret GID without revealing any information about that GID to the AA (preserve the privacy). At the same time, the AA is guaranteed that the agreed upon decryption-keys are the only information that the user learns from the transaction (i.e., provide security by mitigating user collusion). This is derived from blind IBE schemes [15]. We define this algorithm as $[U(params, PK_k, u, GID, decom) \leftrightarrows A_k(params, PK_k, SK_k)] \rightarrow DecryptionKeysforUser$. In this algorithms, the user runs *Commit* and sends the *com* to AA and keeps the *decom* [6]. Then the user and AA interact with each other to generate decryption keys. The decryption keys for the user will be generated only if the output of *Decommit* is 1. From [15], the anonymous key extraction protocol should be leak-free and selective-failure blindness. Definition and security games for leak-free and selective-failure blindness can be found in [6], hence we jump straight into the formulation.

A. Construction of anonymous key issuing protocol

Fig. 3 shows the anonymous key issuing protocol where user with private value $u \in \mathbb{Z}_p$ and A_k with private values $r_{k,u}$, β_k and $\alpha_k \in \mathbb{Z}_p$ jointly computing decryption keys for the user. The decryption keys for u are $D_{k,u}, D^1_{k,u}$ and $D^j_{k,u}$ $\forall a_{k,j} \in \widetilde{A^k_u}$. In order to obtain these keys, first user and A_k interact with each other using two-party protocol (2PC). The 2PC protocol can be done via a general 2PC protocol for a simple arithmetic computation. Alternatively, we can do this more efficiently using the construction in [8]. The 2PC protocol takes $(u, \rho_1, \rho_2) \in \mathbb{Z}_p$ from user and $(r_{k,u}, \beta_k) \in \mathbb{Z}_p$ from A_k and returns $x = (u+\beta_k)\rho_1 \ mod\ p$ and $y = (u+r_{k,u})\rho_2 \ mod\ p$ to A_k. Since, ρ_1 and ρ_2 were randomly generated by user, the AA A_k cannot extract the user identity u from x and y. After executing the 2PC protocol, the user now computes $P = g^{\frac{1}{\rho_1\rho_2}}$, $Q = h^{\frac{1}{\rho_2}}$ and $R = h_1^{\frac{1}{\rho_1}}$ and send those values to A_k. Now A_k computes $\widetilde{D_{k,u}}$, $\widetilde{D^1_{k,u}}$ and $\widetilde{D^j_{k,u}}$ $\forall a_{k,j} \in \widetilde{A^k_u}$ (i.e., randomized decryption credentials) using P, Q, R, x and y and send them back to the user. Now the user exponentiates the obtained values by $\rho_1\rho_2$ to get the decryption keys. Note that, since u coupled non-linearly within the decryption keys. We show in Section III-C that this approach avoid user collusion.

B. Security Analysis

To obtain the decryption credential blindly from A_k $\forall k$, the user needs to prove that he holds the identifier u in zero knowledge. As shown in Fig. 3, the user randomly generates $\rho_1, \rho_2 \in_R \mathbb{Z}_p$ and computes $\Psi_1 = g^{u\rho_1}, \Psi_2 = g^{\rho_1}, \Psi_3 = g^{u\rho_2}$, and $\Psi_4 = g^{\rho_2}$ as commitments. At the end of the 2PC protocol, the AA obtains $x = (r_{k,u} + u)\rho_1 \ mod\ p$, $y = (\beta_k + u)\rho_2 \ mod\ p$. Then AA verifies $g^x \overset{?}{=} AB^{r_{k,u}}$ and $g^y \overset{?}{=} CD^{r_{k,u}}$. If they are correctly verified then the AA continues otherwise it aborts.

Now AA needs to proof that he knows $(\alpha_k, \frac{\beta_k}{x}, \frac{r_{k,u}}{y})$ in zero knowledge to the user. This will be done using the following steps:

1. A_k randomly generates $b_1, b_2, b_3 \in_R \mathbb{Z}_p$, computes $\widetilde{\widetilde{D_{k,u}}} = P^{-b_1}Q^{b_2}R^{b_3}$ and sends $\widetilde{\widetilde{D_{k,u}}}$, and $\widetilde{D_{k,u}}$ to the user
2. User generates $c_1 \in_R \mathbb{Z}_p$ and sends it to the AA
3. AA computes $b'_1 = b_1 - c_1\alpha_k, b'_2 = b_2 - c_1\frac{\beta_k}{x}, b'_3 = b_3 - c_1\frac{r_{k,u}}{y}$ and sends b'_1, b'_2, b'_3 to the user
4. User verifies $\widetilde{\widetilde{D_{k,u}}} \overset{?}{=} P^{-b'_1}Q^{b'_2}R^{b'_3}\widetilde{D_{k,u}}^{c_1}$, otherwise aborts

User $U(params, PK_k, u)$		AA $A_k(params, PK_k, SK_k)$
1. Randomly chooses $u, \rho_1, \rho_2 \in_R \mathbb{Z}_p$ $PoK\{(u, \rho_1, \rho_2) : \Psi_1 = g^{u\rho_1} \bigwedge \Psi_2 = g^{\rho_1} \bigwedge \Psi_3 = g^{u\rho_2} \bigwedge \Psi_4 = g^{\rho_2}\}$	$\xleftrightarrow{2PC}$	$x = (r_{k,u} + u)\rho_1,\ y = (\beta_k + u)\rho_2$
2. Computes $P = g^{\frac{1}{\rho_1\rho_2}},\ Q = h^{\frac{1}{\rho_2}}, R = h_1^{\frac{1}{\rho_1}}$		
		3. $\widetilde{D_{k,u}} = P^{-\alpha_k} Q^{\frac{\beta_k}{x}} R^{\frac{r_{k,u}}{y}}, \widetilde{D^1_{k,u}} = Q^{\frac{1}{x}},$ $\widetilde{D^j_{k,u}} = R^{\frac{p_{k,u}(a_{k,j})}{y t_{k,j}}}, \forall a_{k,j} \in \tilde{A}^k_u$ Let $\sum_2$ denotes the proof: $PoK\left\{(\alpha_k, x, y, \frac{\beta_k}{x}, \frac{r_{k,u}}{y}) : \widetilde{D_{k,u}} = P^{-\alpha_k} Q^{\frac{\beta_k}{x}} R^{\frac{r_{k,u}}{y}} \bigwedge \widetilde{D^1_{k,u}} = Q^{\frac{1}{x}} \bigwedge \widetilde{D^j_{k,u}} = R^{\frac{p_{k,u}(a_{k,j})}{y t_{k,j}}}, \forall a_{k,j} \in \tilde{A}^k_u\right\}$
4. $D_{k,u} = \left(\widetilde{D_{k,u}}\right)^{\rho_1\rho_2},$ $D^1_{k,u} = \left(\widetilde{D^1_{k,u}}\right)^{\rho_1\rho_2},$ $D^j_{k,u} = \left(\widetilde{D^j_{k,u}}\right)^{\rho_1\rho_2}, \forall a_{k,j} \in \tilde{A}^k_u$	$\xleftarrow{\widetilde{D_{k,u}}, \widetilde{D^1_{k,u}}, \widetilde{D^j_{k,u}}, \sum_2}$	

Fig. 3. Anonymous key issuing protocol for decentralized KP-ABE

We ignored the zero-knowledge proofs for $\frac{1}{x}$ and $\frac{1}{y}$ for brevity since they are similar to the above proof.

Theorem 3. *The proposed anonymous key issuing protocol is both leak-free and selective-failure blind.*

Proof. Leak freeness. Suppose there exists an adversary $\mathfrak{U}$ in the real experiment (where $\mathfrak{U}$ is interacting with an honest AA A_k running the anonymous key issuing protocol) and a simulator $\tilde{\mathfrak{U}}$ in the ideal experiment (where $\tilde{\mathfrak{U}}$ can access the trusted AA running the key issuing protocol without privacy preservation) such that no efficient distinguisher $\mathfrak{D}$ can distinguish the real experiment from the ideal experiment. The simulator $\tilde{\mathfrak{U}}$ simulates the communication between the distinguisher $\mathfrak{D}$ and the adversary $\mathfrak{U}$ by passing the input of $\mathfrak{D}$ to $\mathfrak{U}$ and the output of $\mathfrak{U}$ to $\mathfrak{D}$. The simulator $\tilde{\mathfrak{U}}$ works as follows:

1. $\tilde{\mathfrak{U}}$ sends the adversary $\mathfrak{U}$ the public-key PK_k of A_k
2. The adversary $\mathfrak{U}$ must proof the possession of u in zero-knowledge to $\tilde{\mathfrak{U}}$. If proof is successful then $\tilde{\mathfrak{U}}$ obtains (u, ρ_1, ρ_2) using rewind technique
3. $\tilde{\mathfrak{U}}$ sends u to the trusted party. The trusted party runs KeyGen to generates ($D_{k,u}$, $D^1_{k,u}$, $D^j_{k,u}$) and responds to $\tilde{\mathfrak{U}}$
4. Now $\tilde{\mathfrak{U}}$ computes $\left(D_{k,u}{}^{\frac{1}{\rho_1\rho_2}}, D^1_{k,u}{}^{\frac{1}{\rho_1\rho_2}}, D^j_{k,u}{}^{\frac{1}{\rho_1\rho_2}}\right)$ and sends them to $\mathfrak{U}$

If ($D_{k,u}$, $D^1_{k,u}$, $D^j_{k,u}$) are correct keys from the trusted AA in the ideal experiment, then $\left(D_{k,u}{}^{\frac{1}{\rho_1\rho_2}}, D^1_{k,u}{}^{\frac{1}{\rho_1\rho_2}}, D^j_{k,u}{}^{\frac{1}{\rho_1\rho_2}}\right)$ are the correct keys from A_k in the real experiment. Hence, ($D_{k,u}$, $D^1_{k,u}$, $D^j_{k,u}$) and $\left(D_{k,u}{}^{\frac{1}{\rho_1\rho_2}}, D^1_{k,u}{}^{\frac{1}{\rho_1\rho_2}}, D^j_{k,u}{}^{\frac{1}{\rho_1\rho_2}}\right)$ are correctly distributed and no efficient distinguisher can distinguish the real experiment with the ideal experiment.

Proof. Selective-failure blindness. The adversarial AA A_k submits the public key PK_k, and two GIDs u_0 and u_1. Then, a bit $b \in \{0, 1\}$ is randomly selected. A_k can have a black box access to u_0's and u_1's parameters i.e., $U(params, PK_k, u_b)$ and $U(params, PK_k, u_{b-1})$. Then, U executes the anonymous key issuing protocol with A_k and outputs secret keys for u_b and u_{1-b} i.e., SK_{u_b} and $SK_{u_{1-b}}$.

1. If $SK_{u_b} \neq \perp$ and $SK_{u_{1-b}} \neq \perp$ then A_k is given (SK_{u_b}, $SK_{u_{1-b}}$)
2. If $SK_{u_b} \neq \perp$ and $SK_{u_{1-b}} = \perp$ then A_k is given (ϵ, $\perp$)
3. If $SK_{u_b} = \perp$ and $SK_{u_{1-b}} \neq \perp$ then A_k is given ($\perp$, ϵ)
4. If $SK_{u_b} = \perp$ and $SK_{u_{1-b}} = \perp$ then A_k is given ($\perp$, $\perp$)

At the end A_k submits his prediction on b.

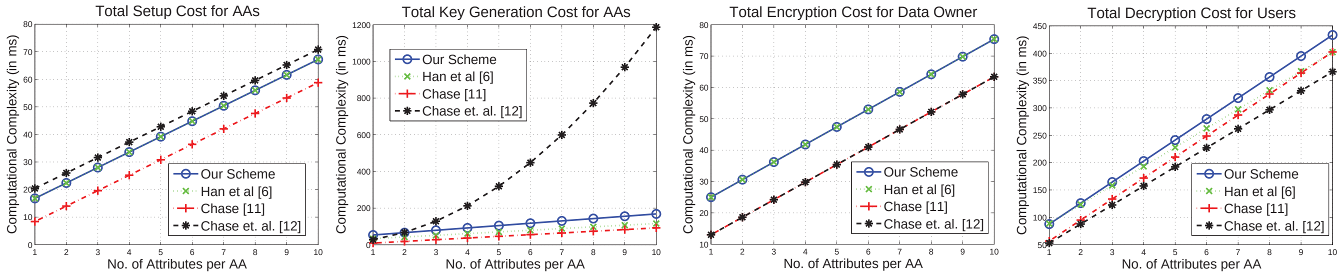
In the anonymous key issuing protocol, U sends four random parameters $\Psi_1 = g^{u_b\rho_1} \in \mathbb{G}_1, \Psi_2 = g^{\rho_1} \in \mathbb{G}_1, \Psi_3 = g^{u_b\rho_2} \in \mathbb{G}_1$, and $\Psi_4 = g^{\rho_2} \in \mathbb{G}_1$ to the adversarial AA A_k and proves $PoK(u_b, \rho_1, \rho_2)$. Now it is A_k's turn to respond. So far, A_kï£¡s view on the two black boxes is computationally undistinguishable. Otherwise, the hiding property of the commitment scheme and the witness undistinguishable property of the zero-knowledge proof will be broken. Suppose that A_k uses any computing strategy to output secret keys $\{D_{k,u_b}, D^1_{k,u_b}, D^j_{k,u_b} \forall a_{k,j} \in \tilde{A}^k_{u_b}\}$ for the first black box. In the following, we will show that A_k can predict SK_{u_b} of U without interacting with the two black boxes:

1. A_k checks
   $$PoK\left\{(\alpha_k, \frac{\beta_k}{x}, \frac{r_{k,u}}{y}) : \widetilde{D_{k,u}} = P^{-\alpha_k} Q^{\frac{\beta_k}{x}} R^{\frac{r_{k,u}}{y}} \bigwedge \widetilde{D^1_{k,u}} = Q^{\frac{1}{x}} \bigwedge \widetilde{D^j_{k,u}} = R^{\frac{p_{k,u}(a_{k,j})}{y t_{k,j}}}, \forall a_{k,j} \in \tilde{A}^k_u\right\}$$
   If proof fails, A_k sets $SK_0 = \perp$
2. A_k generates different secret keys $\{D_{k,u_b}, D^1_{k,u_b}, D^j_{k,u_b} \forall a_{k,j} \in \tilde{A}^k_{u_b}\}$ for the second black box and a proof of knowledge:
   $$PoK\left\{(\alpha_k, \frac{\beta_k}{x}, \frac{r_{k,u}}{y}) : \widetilde{D_{k,u}} = P^{-\alpha_k} Q^{\frac{\beta_k}{x}} R^{\frac{r_{k,u}}{y}} \bigwedge \widetilde{D^1_{k,u}} = Q^{\frac{1}{x}} \bigwedge \widetilde{D^j_{k,u}} = R^{\frac{p_{k,u}(a_{k,j})}{y t_{k,j}}}, \forall a_{k,j} \in \tilde{A}^k_u\right\}$$
   If proof fails, A_k sets $SK_1 = \perp$
3. Finally A_k outputs his prediction on (u_0, u_1) with (SK_{u_b}, $SK_{u_{1-b}}$) if $SK_{u_b} \neq \perp$ and $SK_{u_{1-b}} \neq \perp$; (ϵ, $\perp$) if $SK_{u_b} \neq \perp$ and $SK_{u_{1-b}} = \perp$; ($\perp$, ϵ) if $SK_{u_b} = \perp$ and $SK_{u_{1-b}} \neq \perp$; ($\perp$, $\perp$) if $SK_{u_b} = \perp$ and $SK_{u_{1-b}} = \perp$.

The predication on (u_0, u_1) is correct, and has the identical distribution with the black box. Because A_k performs the same check as the honest U, it outputs the valid keys as U obtains from anonymous key issuing protocol. Hence, if A_k can predict the final outputs of the two black boxes, the advantage of A_k in distinguishing the two black boxes is the same without the final outputs. Therefore, the advantage of A_k should come from the received $\Psi_1, \Psi_2, \Psi_3, \Psi_4 \in \mathbb{G}_1$ and $PoK(u, \rho_1, \rho_2)$. From the hiding property of the commitment scheme and witness undistinguishable property of the zero-knowledge proof, A_k cannot distinguish one from the other with nonnegligible advantage. Hence our anonymous key issuing protocol is secure against selective-failure.

C. User collusion

In this section we demonstrate the user collusion vulnerability in [6] identified by [7]. We will use a simple example for this, however, a generic scenario is provided in [7]. Later we show



(a) (b) (c) (d)

Fig. 4. Comparison of computational costs of four different schemes against four different steps.

that the proposed scheme is free from the identified vulnerability and discuss the reason. Lets consider a scenario with two AAs A_1 and A_2. Let A_1 monitors attribute $\{a_{1,1}\}$ and A_2 monitors $\{a_{2,1}\}$. Hence according to the original scheme presented in Fig. 1, the public keys and secret keys of A_k are $PK_k = \{Y_k = e(g,g)^{\alpha_k}, Z_k = g^{\beta_k}, T_{k,j} = g^{t_{k,1}}\}$ and $SK_k = \{\alpha_k, \beta_k, t_{k,1}\}$ for $k = \{1,2\}$. Suppose the encryption algorithm specifies the attribute set for message m is $\tilde{A}_m = \{a_1, a_2\}$ while the users U_1 and U_2 with identifiers u_1 and u_2 only have the decryption credential for the attribute $\{a_1\}$ and the user U_3 with identifier u_3 only has the credential for attribute $\{a_2\}$. Note that none of these three users can decrypt the ciphertext alone. However, due to the vulnerability in [6], these users can collude together and generate decryption credentials to decrypt the ciphertext encrypted by attribute set $\tilde{A}_m = \{a_1, a_2\}$ as follows: according to Fig. 1, credential issued to U_1 and U_2 are

$$\begin{aligned} D_{1,u_1} &= g^{\alpha_1} h^{r_{1,u_1}} h_1^{u_1\beta_1}, D_{1,1} = h^{\frac{r_{1,u_1}}{t_{1,1}}}, \\ D_{1,u_2} &= g^{\alpha_1} h^{r_{1,u_2}} h_1^{u_2\beta_1}, D_{2,1} = h^{\frac{r_{1,u_2}}{t_{1,1}}}, \end{aligned}$$

U_1 and U_2 can collude to compute the secret key corresponding to the user U_3 as follows:

$$\begin{aligned} D_{1,u_3} &= D_{1,u_1}.\left(\frac{D_{1,u_1}}{D_{1,u_2}}\right)^{\frac{u_3-u_1}{u_1-u_2}} \\ &= g^{\alpha_1} h^{r_{1,u_1} + \frac{(r_{1,u_1}-r_{1,u_2})(u_3-u_1)}{u_1-u_2}} h_1^{u_3\beta_1} = g^{\alpha_1} h^{r_{1,u_3}} h_1^{u_3\beta_1}, \\ D_{3,1} &= D_{1,1}.\left(\frac{D_{1,1}}{D_{2,1}}\right)^{\frac{u_3-u_1}{u_1-u_2}} \\ &= h^{\frac{r_{1,u_1}}{t_{1,1}} + \frac{(r_{1,u_1}-r_{1,u_2})(u_3-u_1)}{(u_1-u_2)t_{1,1}}} = h^{\frac{r_{1,u_3}}{t_{1,1}}}. \end{aligned}$$

Note that D_{1,u_3} and $D_{3,1}$ are valid credentials for U_3 for attribute $a_{1,1}$ since the power of h_1 (i.e., $u_3\beta_1$) and power of h_1 (i.e., random value $r_{1,u_1} + \frac{(r_{1,u_1}-r_{1,u_2})(u_3-u_1)}{u_1-u_2}$) are correctly produced in the credentials D_{1,u_3} and $D_{3,1}$ due to the collusion.

Let us analyse the reason behind this successful collusion. If we look at the variables (i.e., β_1, and u_3) in the power of h_1 in D_{1,u_3}, β_1 is known to AA and u_3 are known to the user. However, the power, $\beta_1 u_3$ is just a multiplication (it is very naive) of both the variables. Hence the users can remove u_1 and u_2 while keeping β_1 using the above simple algebraic operations to produce decryption credential for U_3.

In the proposed scheme, if we compute $D_{1,u_1}.\left(\frac{D_{1,u_1}}{D_{1,u_2}}\right)^{\gamma}$ then the power of the generator h becomes $\frac{\beta_1}{r_{1,u_1}+u_1} + \gamma\left(\frac{\beta_1}{r_{1,u_1}+u_1} - \frac{\beta_1}{r_{1,u_2}+u_2}\right)$, and the power of generator h_1 becomes $\frac{r_{1,u_1}}{\beta_1+u_1} + \gamma\left(\frac{r_{1,u_1}}{\beta_1+u_1} - \frac{r_{1,u_2}}{\beta_1+u_2}\right)$ where γ could be any scalar (i.e., $\frac{u_3-u_1}{u_1-u_2}$). It is obvious from both the powers, even though users know u_1 and u_2, they cannot remove the u_1 and u_2 since they are coupled non-linearly with r_{1,u_1}, r_{1,u_2} and β_1 which are known only to the AA (i.e., $\frac{1}{r_{1,u_1}+u_1}$, $\frac{1}{\beta_1+u_1}$). Hence, u_1 and u_2 cannot be replaced with u_3 without knowing β_1, r_{1,u_1} and r_{1,u_2} . Hence our method withstands the collusion found in [6]. In general, since the variable known to user and the variables known to the AA are non-linearly coupled in our scheme, it is impossible for two users to generate decryption credential for third user without involving AA.

IV. Complexity Comparison

In this section, we compare the computational complexity of our scheme against the following KP-ABE schemes: Han et. al decentralized KP-ABE scheme [6], Chase MA-ABE KP-ABE scheme with central AA [11], and Chase et.al MA-ABE KP-ABE scheme without central AA [12]. Let us assume there are N number of AAs and each AA monitors n number of attributes. In order to compare the complexity at worst case scenario, let us assume that the ciphertext encrypted using all the attributes in the system (i.e., nN) and user has decryption credentials for all the attributes. Let us denote the computational time (in ms) for one multiplication, one exponentiation, and one pairing as C_m, C_e, and C_p, respectively.

In order to simulate, we use the benchmark time values given in the popular pairing-based cryptography library namely jPBC in [16]. Table I shows the time values (in ms) for C_m, C_e, and C_p for two different testbeds: testbed 1 uses Intel(R) Core(TM) 2 Quad CPU Q6600 with 2.40GHz and 3 GB memory running on Ubuntu 10.04 and testbed 2 uses HTC Desire HD A9191 smart phone running on Android 2.2. The time values given in Table I are for a symmetric elliptic curve called a-curve, where the base field size is 512-bit and the embedding degree is 2. The a-curve has a 160-bit group order.

We consider the healthcare scenario described in Section I as an example to simulate the comparison of four different schemes. Without loss of generality lets assume $N = 2$ and n varies from 1 to 10. As shown in Figures 1 and 2, each algorithm consists of four sub algorithms. Fig. 4 compares the performance of four schemes across four sub algorithms. In Fig. 4(a), the time complexity for setting up the ABE system for all four algorithms increase linearly with same order against the number of attributes per AA. Setup cost for [11] is better than other schemes. It is due to a simplest problem formulation without privacy consideration (i.e., central AA knows secret keys of all AAs).

In Fig. 4(b), the key generation time complexity for [12] increases quadratically with number of attributes due to the interaction between pair of AAs to agree on a shared secret per

TABLE I

Time complexity measures for two different testbeds.

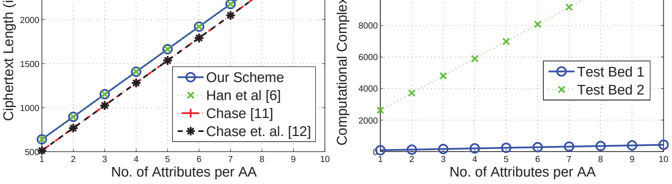
	Testbed 1 (ms)	Testbed 2 (ms)
C_p	14.6	491.2
C_e	2.8	34.1
C_m	1.8	20

attribute. However, complexity of other schemes increase linearly with almost same order. Complexity of the proposed scheme is high when $n = 1$ due to generation of other independent credential $D_{k,u}$. Han et. al. scheme performs better than the proposed scheme due the difference in number of credentials generated in both the schemes.

In Fig. 4(c), the time complexity for encryption increases linearly with same order for all schemes similar to the setup. Han et. al's scheme and the proposed scheme show equal performance while other two schemes also show equal performance. This clearly shows that there is no significant difference in Encryption sub algorithm and new schemes are built on top of the previous schemes.

Finally the time complexity of decryption shown in Fig. 4(d) is slightly different compared to other three sub algorithms. Even though the time complexity of all four algorithms increase linearly, the orders are different for all four algorithms. In fact Han et. al perform better than [12] after $n = 10$. The proposed algorithm takes longer (roughly $10ms$ longer for 10 attributes) to decrypt messages than other algorithms. This is solely due to the additional operations involved for the privacy protection. In overall, Chase's scheme [11] performs better than other three schemes due to it's simplicity. However, the algorithms in [11], [12] need a central AA and/or interaction among AAs; hence [11], [12] are vulnerable for single point of failure. The proposed scheme and Han et. al's scheme are performing equally well as each other. However, the proposed algorithm mitigates the user collusion vulnerability compared to [6]. Due to page limitation, we restricted to $N = 2$ case, however we observed the similar trend for $N > 2$ cases.

Let us now compare the ciphertext length and performance of the proposed scheme on mobile platform. Fig. 5(a) shows ciphertext length for all schemes against the number of attributes. The ciphertext length is increasing linearly with number of attributes at same order for all schemes. This is one of the drawbacks of these schemes; hence, developing a novel scheme which ensures the user's privacy with constant size ciphertext could be a potential future work. Recent trend of Internet of Things replaces PCs with smart devices and it is anticipated that decryption part of the proposed scheme could be done on mobile device i.e., users want to download and read data on their smartphones. To compare the performance on smartphones against PC, we simulate the time complexity of decryption sub algorithm on both the test beds given in Table I. As expected, Fig. 5(b) shows that the decryption process on smartphone is much slower than PC. More importantly, even though the time complexity on both the test beds are increasing linearly, the complexity on test bed 2 increases with higher order than PC. The main reason behind this is that the mobile device used in test bed 2 is equipped with one processor. This led the non-linear relationship of C_p between both the test beds (in test bed 1, $C_p \cong 5C_m$ but in test bed 2 $C_p \cong 14C_m$).



(a) (b)

Fig. 5. (a) Ciphertext length of four different schemes, (b) decryption complexity of the proposed scheme is two different test beds.

V. Conclusions

In this paper, we proposed a PP KP-ABE scheme for a distributed data sharing environment. The proposed scheme enables users to download and decrypt the data from online such as cloud without revealing their attributes to the third-parties. The novelty of the work is to mitigate the user collusion attack in the existing scheme. We used anonymous key issuing protocol to strengthen the bind between user identity and decryption keys; hence, two or more users cannot pool their keys to generate decryption keys for an unauthorized user. We validated the security of the proposed scheme using the decisional bilinear Diffie-Hellman standard complexity assumption.

References

[1] A. Sahai, and B. Waters, Fuzzy Identity-Based Encryption. Advances in Cryptology EUROCRYPT, vol. 3494, pp. 557, 2005.

[2] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based Encryption for Fine-Grained Access Control of Encrypted Data. In Proc. 13th ACM conf. Comp. and Commun. security (CCS '06), New York, USA, pp. 89-98 2006.

[3] F. Li, Y. Rahulamathavan, M. Rajarajan, R. C.-W Phan. Low complexity multi-authority attribute based encryption scheme for mobile cloud computing. In Proc. IEEE 7th Int'l Symp. Service Oriented System Engineering (SOSE), San Francisco, USA, pp. 573–577, Mar. 2013.

[4] F. Li, Y. Rahulamathavan, M. Rajarajan, R. C.-W Phan. LSD-ABAC: Lightweight Static and Dynamic Attributes Based Access Control Scheme for Secure Data Access in Mobile Environment. In Proc. IEEE Local Computer Networks (LCN), Edmonton, Canada, Sept. 2014.

[5] J. Bethencourt, A. Sahai, B. Waters. Ciphertext-Policy Attribute-Based Encryption. In IEEE Symposium on Security and Privacy, SP 07, pp. 321–334, May 2007.

[6] J. Han, W. Susilo, Y. Mu, and J. Yan: Privacy-Preserving Decentralized Key-Policy Attribute-Based Encryption. In IEEE Trans. Parallel and Distributed Systems, vol. 23, no. 11, pp. 2150–2162, Nov. 2012.

[7] A. Ge, J. Zhang, R. Zhang, C. Ma, and Z. Zhang: Security Analysis of a Privacy-Preserving Decentralized Key-Policy Attribute-Based Encryption Scheme. In IEEE Trans. Parallel and Distributed Systems, vol. 24, no. 11, pp. 2319–2321, 2013.

[8] M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Lysyanskaya, and H.Shacham. Randomizable Proofs and Delegatable Anonymous Credentials. In Advances in Cryptology-CRYPTO 2009, pp. 108–125. Springer Berlin Heidelberg, 2009.

[9] A. B. Lewko and B. Waters. Decentralizing attribute-based encryption. In EUROCRYPT, ser. Lecture Notes in Computer Science, K. G. Paterson, Ed., vol. 6632. Springer, 2011, pp. 568-588.

[10] B. Waters. Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient, and Provably Secure Realization. In Proc. 14th Intï£¡l Conf. Practice and Theory in Public Key Cryptography Conf. Public Key Cryptography (PKC ï£¡11), D. Catalano, N. Fazio, R. Gennaro, and A. Nicolosi, eds., pp. 53-70, Mar. 6-9 2011.

[11] M. Chase, Multi-authority Attribute Based Encryption, In LNCS, Berlin Heidelberg, pp. 515–534, vol. 4392, 2007.

[12] M. Chase and Sherman S.M. Chow. Improving privacy and security in multi-authority attribute-based encryption. In Proc. 16th ACM conf. Comp. and Commun. Security (CCS '09), New York, USA, pp. 121-130, 2009.

[13] Stefan Brands. Rethinking Public Key Infrastructure and Digital Certificates ï£¡ Building in Privacy. PhD thesis, Eindhoven Inst. of Tech. 1999.

[14] J. Camenisch and A. Lysyanskaya. Efficient Non-transferable Anonymous Multi-show Credential System with Optional Anonymity Revocation. In EUROCRYPT 2001, vol. 2045 of LNCS, pp. 93–118. Springer Verlag, 2001.

[15] J. Camenisch, M. Kohlweiss, A. Rial, and C. Sheedy. Blind and anonymous identity-based encryption and authorised private searches on public key encrypted data. In Proc: PKCï£¡09, volume 5443 of LNCS, pp. 196–214. Springer, 2009.

[16] The Java Pairing-Based Cryptography Library (JPBC), http://tinyurl.com/ll2p39t

[17] A. Abbas, S. U. Khan. A Review on the State-of-the-Art Privacy-Preserving Approaches in the e-Health Clouds. IEEE Journala of Biomedical and Health, vol.18, no.4, pp.1431-1441, July 2014.

[18] Z. M. Fadlullah, N. Kato, R. Lu, X. Shen, and Y. Nozaki, "Towards Secure Targeted Broadcast in Smart Grid", IEEE Communications Magazine, vol. 50, No. 5, pp. 150-156, 2012.