Lightweight attribute-based encryption supporting access policy update for cloud assited IoT

Sana Belguith, Nesrine Kaaniche, Giovanni Russello

Lightweight Attribute-Based Encryption Supporting Access Policy Update for Cloud Assitsed IoT

Sana Belguith[1], Nesrine Kaaniche[2], Giovanni Russello[1]
[1]*Cyber Security Foundry, University of Auckland, New Zealand*
[2]*SAMOVAR, CNRS, Telecom SudParis, University Paris-Saclay, Paris, France*
sbel452@aucklanduni.ac.nz,Nesrine.Kaaniche@telecom-sudparis.eu,g.russello@auckland.ac.nz

Keywords: Constant-size Attribute based encryption, Access policy update, Cloud assisted IoT, Confidentiality, Access control.

Abstract: Cloud-assisted IoT applications are gaining an expanding interest, such that IoT devices are deployed in different distributed environments to collect and outsource sensed data to remote servers for further processing and sharing among users. On the one hand, in several applications, collected data are extremely sensitive and need to be protected before outsourcing. Generally, encryption techniques are applied at the data producer side to protect data from adversaries as well as curious cloud provider. On the other hand, sharing data among users requires fine grained access control mechanisms. To ensure both requirements, Attribute Based Encryption (ABE) has been widely applied to ensure encrypted access control to outsourced data. Although, ABE ensures fine grained access control and data confidentiality, updates of used access policies after encryption and outsourcing of data remains an open challenge. In this paper, we design PU-ABE, a new variant of key policy attribute based encryption supporting efficient access policy update that captures attributes addition and revocation to access policies. PU-ABE contributions are multifold. First, access policies involved in the encryption can be updated without requiring sharing secret keys between the cloud server and the data owners neither re-encrypting data. Second, PU-ABE ensures privacy preserving and fine grained access control to outsourced data. Third, ciphertexts received by the end-user are constant sized and independent from the number of attributes used in the access policy which affords low communication and storage costs.

1 INTRODUCTION

Nowadays, IoT applications are widely used in different fields such as smart cities, e-health, intelligent transport systems, to name a few (Farhan et al., 2018; Atwady and Hammoudeh, 2017; Coates et al., 2017; Belguith et al., 2018). Due to their limited storage and computation capacities, IoT devices are usually assisted with cloud services to store and process generated data (Kaaniche and Laurent, 2017b; Belguith et al., 2015). IoT devices produce huge amounts of data which need to be securely collected and shared among different users (Bacis et al., 2016a; Bacis et al., 2016b). Consequently, encryption and access control mechanisms are important to protect data from unauthorised access (Belguith et al., 2016).

Attribute based encryption (ABE) ensures encrypted access control to outsourced data while limiting privacy leakage of data producers and users (Kaaniche and Laurent, 2017a). ABE consists of enciphering data with respect to an access policy over a set of attributes where users who have the matching attributes can recover data. Nowadays, with the emergence of applications adapted to distributed and dynamic environments, several settings require adding new users, strengthening access patterns and/or removing current users from systems. A naive solution is to re-encrypt the whole data contents using new access policy by the data owner and re-outsource them to the cloud server. However, this solution is extremely expensive in computation and communications costs as it should be re-executed at every time a user is added or removed. Proxy re-encryption techniques are used to update users in outsourced systems. These mechanisms allow a server to re-encrypt stored data without accessing their content using a re-encryption key (Liang et al., 2015; Ge et al., 2018). Attribute-based Proxy Re-encryption (AB-PRE) systems have been applied for access policy update. In AB-PRE, when the data owner decides to update access policies of some ciphertexts, she uses her private key to generated a re-encryption key for each ci-

phertext to be changed from an old access policy to a new one. Afterwards, all the generated re-encryption keys are uploaded to a proxy server to update the ciphertext using the received keys. Although AB-PRE schemes allow re-encryption without decrypting ciphertext or accessing the plaintexts, it requires that the data owner generates valid re-encryption keys. When the number of ciphertext rises, it becomes inefficient for a data owner to generate all the re-encryption keys and upload them to the proxy. Furthermore, this may also be unfeasible for limited bandwidths. Therefore, attribute-based proxy re-encryption schemes may not be efficient when updating a huge the number of ciphertexts.

Key Policy Attribute Based Encryption (KP-ABE) is widely applied to secure data in several distributed systems such as Publish and Subscribe systems (Pub/Sub) (Ion et al., 2012; Esposito and Ciampi, 2015), pay-TV systems (Ogawa et al., 2017), vehicular networks (Nkenyereye et al., 2016), ..., where rules on who may read a document must be specified but it is unable to specify policies on a per-message basis. Obviously, these dynamic environments usually require efficiently adding new users and/or revoking existent users. That is, KP-ABE consists in labeling user's key by an access structure that specifies which type of ciphertext the key can decrypt, while ciphertext are labeled by a set of attributes. Thus, KP-ABE are adapted to distributed and decentralized environments. Instead, Ciphertext Policy Attribute Based Encryption (CP-ABE) schemes consists in associating an access structure to the ciphertext while assigning a set of credentials to deciphering users. CP-ABE schemes supporting policy update have been recently explored by Jiang et al. (**?**; Jiang et al., 2017). Indeed, the authors proposed a CP-ABE scheme where new attributes can be added or current attributes can be removed efficiently without sharing re-encryption keys.

This motivates us to introduce the first KP-ABE scheme supporting adding and/or removing attributes from the access policy without sharing keys with the cloud server.

Contributions – In this paper, we propose Policy Update Attribute Based Encryption (PU-ABE), a novel key policy attribute based encryption scheme which supports access policy update. This proposed scheme is suitable for bandwidth-limited applications as the size of the ciphertext received by end-users does not depend on the number of attributes involved in the access policy. In PU-ABE, the encrypting entity generates a ciphertext involving encrypted data together with some extra components used for supporting the access policy update feature. The ciphertext is forwarded to the cloud server that can update the access policy upon demand. Indeed, the cloud server does not need to be trusted. That is, it stores and shares ciphertexts among authorised users and also executes access policy update algorithm as requested. While executing these functionalities, the remote server is unable of decrypting any ciphertexts neither accessing any secret keys. For supporting the policy update feature, PU-ABE provides two functions. The first function permits to add new attributes to the access policy used to encrypt data. The second function ensures revoking attributes from the access policy used in the encryption phase. To prove the security of our proposed constructions, we present an updated security model to capture security requirements related to policy updates.

Paper Organisation – The remainder of this work is as follows: Section 2 highlights security considerations and design goals. Then, Section 3 reviews related work and section 4 describes the system and security models. Section 5 presents the mathematical background. In Section 6, an overview of PU-ABE is introduced and the detailed construction are presented. Section 7 presents a rigorous security discussion. Finally, a theoretical analysis of computational performances is presented in Section 8, before concluding in Section 9.

2 PU-ABE FRAMEWORK

In this section, we first present the network model, detailing the involved entities and their interactions in Section 2.1. Then, we detail the security requirements that the proposed system should fulfill in Section 2.2.

2.1 Architecture

As depicted by Figure 1, our PU-ABE framework considers a cloud service provider that stores data generated by data owners and share them among authorised users. Four different entities are defined as follows:

- The Central Trusted Authority (CTA), known by the Attribute authority, is responsible for generating the global public parameters and issuing users' secret keys. CTA is considered as a trusted entity in our model.
- The Cloud Service Provider (CSP) is a remote cloud server who stores and shares data among authorised users. CSP is also responsible of executing the update algorithm to change the access

policy involved in the ciphertext, w.r.t. the data owner's recommendations.

- The data owner (O) is the data producer. She defines access rights and encrypts data with respect to them before outsourcing to the cloud. In addition, the data owner generates extra ciphertext components used by the update algorithm.
- The data users (U) requests access to outsourced data. She decrypts the received data using his access rights. A user may be malicious if she tries to access data without authorisation.

2.2 Security Requirements

To design an efficient attribute based encryption scheme supporting efficient access policy update, the following requirements need to be achieved:

- ***access policy update*** – our PU-ABE scheme should ensure adding new attributes and/or removing attributes from the access policy.
- ***flexible access control*** – our proposal should ensure flexible security policies among dynamic groups of users, w.r.t. forward secrecy and backward secrecy.
  - backward secrecy means that a new added user to a group is unable to decrypt information created prior to their introduction.
  - forward secrecy means that a compromise of the secret key does not affect the secrecy of future encrypted data.
- ***data confidentiality*** – our PU-ABE scheme has to protect the secrecy of outsourced and encrypted data contents against curious users and curious cloud service provider.
- ***low computation overhead and storage cost*** – the proposed algorithms should have low processing complexity and acceptable storage cost to be adapted to resource-constrained devices and distributed environments.

3 ATTRIBUTE BASED ENCRYPTION SCHEMES

Attribute-based Encryption (ABE) has been designed to ensure encrypted flexible access control for outsourced data (Sahai and Waters, 2005). Unlike traditional public key encryption schemes, ABE consists in encrypting data for many users. Therefore, decrypting entities' private keys and encrypted data are labeled with a set of attributes or a structure over attributes. A user is able to decrypt a ciphertext if there is a match between her private key and the ciphertext (Bethencourt et al., 2007). Attribute based encryption schemes are classified into two categories, namely: Key-Policy ABE (KP-ABE) and Ciphertext-Policy ABE (CP-ABE) (Goyal et al., 2006). In KP-ABE, ciphertexts are labeled with a set of attributes while users' private keys are associated with an access policy which can be any monotonic structure. The user is able to decrypt the ciphertext if her access policy is satisfied by the attributes embedded in the ciphertext. KP-ABE schemes have been widely applied to secure data in distributed systems such as Internet of Things, publish and subscribe systems, intelligent transport systems, etc. (Yao et al., 2015).

Although ABE schemes ensure flexible access control to encrypted data, the communication and computation overhead as well as the bandwidth consumption increase exponentially with the number of attributes required in the access policies. To save the storage cost of ciphertext and processing overhead of encryption, attribute based encryption schemes with constant ciphertext size have been introduced (Herranz et al., 2010; Ge et al., 2012; Attrapadung et al., 2012; Wang and Luo, 2012). In these schemes, the size of the generated ciphertext does not depend on the number of attributes used on the threshold access policies, which presents an interesting feature mainly for resource-constrained devices. Herranz et al. (Herranz et al., 2010) have proposed the first constant size threshold ciphertext-policy attribute based encryption scheme. Indeed, the ciphertext size is constant and does not depend on the number of attributes involved in the threshold access policies. Afterwards, several CP-ABE schemes with constant cipheretxt size have been proposed (Ge et al., 2012; Belguith et al., 2017; Li et al., 2017). Due to the construction of CP-ABE schemes, monotone access policies based schemes can not be extended to ensure a constant ciphertext size. For instance, these schemes consist in only using threshold or conjunctive access policies which do not provide the desired expressiveness.

Several expressive Key policy attribute based encryption schemes with constant ciphertext size have been designed (Attrapadung et al., 2012; Wang and Luo, 2012; Emura et al., 2009). Wang et al. (Wang and Luo, 2012) have proposed a KP-ABE scheme with constant ciphertext size. This scheme relies on a monotone access policy to express the users' attributes.

Although, these schemes ensure reduced communication and computation costs, they still present a major limitation which is their incapacity of changing access policies of ciphertexts. In dynamic envi-

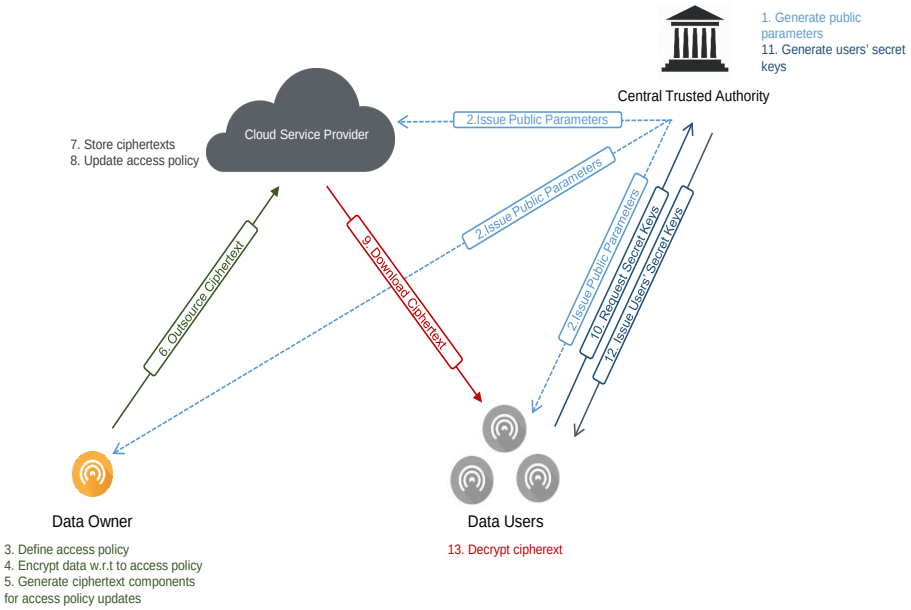

Figure 1: PU-ABE involved entities and their interaction

ronments, users may be often added or removed, then access policies should be updated to support these changes. Recently, the first CP-ABE with policy update has been proposed by Jiang et al. (**?**; Jiang et al., 2017). The authors introduced a new variant of CP-ABE supporting access policy update that captures the functionalities of attribute addition and revocation from access policies. They provide two CP-ABE schemes supporting AND-gate access policies with constant-size ciphertexts.

PUABE

4 MODEL DESCRIPTION

In this section, we first present the system model of our PU-ABE scheme (subsection 4.1). Then, we detail our security model (subsection 4.2).

4.1 System Model

Our policy-update key-policy attribute based encryption scheme consists of five randomized algorithms: setup, encrypt, update, keygen and decrypt, defined as follows:

$\mathsf{setup}(\xi) \rightarrow (\mathsf{pp}, \mathtt{msk})$ – the setup algorithm is performed by a central trusted authority, known by the attribute authority. It takes as input a security parameter ξ and outputs the public parameters pp and the secret master key msk.

$\mathsf{encrypt}(\mathsf{pp}, \mathcal{S}, M) \rightarrow CT$ – the encryption algorithm is performed by an enciphering entity $\mathcal{E}$. It takes as inputs the public parameters pp, the set of the encryption attributes $\mathcal{S}$ and the message M. This algorithm outputs the encrypted message, referred to as CT.

$\mathsf{update}(\mathsf{pp}, CT, \mathsf{ind}, \mathcal{U}) \rightarrow CT'$– the update algorithm is executed by a cloud server. It takes as inputs the public parameters pp, a ciphertext CT that contains the set of enciphering attributes $\mathcal{S} = \{a_i\}_{i=1..m}$ such that $|\mathcal{S}| = m$, an operation indicator ind where ind = add or ind = revoke and a set of attributes $\mathcal{U}$ with $\mathcal{U} \cap \mathcal{S} = \emptyset$ if ind = add or $\mathcal{U} \subset \mathcal{S}$ if ind = revoke. It outputs a new ciphertext CT' for the new encrypting set of attributes $\mathcal{S}'$ such as $\mathcal{S}' = \mathcal{S} \cup \mathcal{U}$ or $\mathcal{S}' = \mathcal{S} \setminus \mathcal{U}$ w.r.t. ind value.

$\mathsf{keygen}(\mathsf{pp}, \mathtt{msk}, \Psi) \rightarrow sk$ – this randomized algorithm is executed by the attribute authority to derive the secret keys of a decrypting entity $\mathcal{D}$. Given the public parameters pp, an access policy Ψ of the user $\mathcal{D}$ and the secret master key msk. The algorithm outputs the user's secret key sk w.r.t. Ψ.

$\mathsf{decrypt}(\mathsf{pp}, sk, CT') \rightarrow M$ – the decryption algorithm is executed by the decrypting entity $\mathcal{D}$. It takes as inputs the public parameters pp, the user's secret key sk and the ciphertext CT'. The algorithm returns the message M if $\mathcal{D}$ has correctly obtained the secret key related to the required set of attributes for deciphering the encrypted message. Otherwise, the algorithm outputs a reject symbol $\perp$.

Our PU-ABE scheme has to satisfy the **correctness property**. The correctness property requires that for all security parameter ξ, all attribute uni-

verse descriptions $\mathbb{U}$, all $(\mathtt{pp},\mathtt{msk}) \in \mathsf{setup}(\xi)$, all $(\mathcal{S},\mathcal{U}) \subseteq \mathbb{U}$ (i.e; $\mathbb{U}$ is the attribute universe), all $sk \in \mathsf{keygen}(\mathtt{pp},\mathtt{msk},\Psi)$, all $M \in \mathbb{M}$ (i.e; $\mathbb{M}$ is the message universe), all $\Psi \in \mathcal{G}$ ($\mathcal{G}$ is the access policy space), all $CT \in \mathsf{encrypt}(\mathtt{pp},\mathcal{S},M)$, and all $CT' \in (\mathtt{pp},CT,\mathsf{ind},\mathcal{U})$ if the decrypting user has correctly obtained the secret key sk related to the Ψ required access policy $\mathcal{S}'$ for deciphering the encrypted message, the $\mathtt{derypt}(\mathtt{pp},sk,CT')$ outputs M.

4.2 Security Model

For designing a secure policy-update attribute based encryption scheme, we consider the case of malicious adversaries with respect to the indistinguishability property. The indistinguishability property means that if an adversary has some information about the plaintext, he should not learn about the ciphertext. This security notion requires the computational impossibility to distinguish between two messages chosen by the adversary with a probability greater than a half.

To design the most suitable security model considering the confidentiality requirement, we adopt an updated security model to capture security requirements related to policy updates (Jiang et al., 2017). PU-ABE is said to be indistinguishable against non-adaptive chosen ciphertext attacks if there is no probabilistic polynomial time (PPT) adversary that can win the Exp^{conf} security game with non-negligible advantage. The Exp^{conf} game is formally defined, between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ as follows:

INITIALISATION – $\mathcal{A}$ selects a set of encryption attributes $\mathcal{S}^*$ (i.e; $\mathcal{S}^*$ corresponds to the set of attributes specified for the encryption) to be used for encrypting the challenge ciphertext, as a set of attributes $\mathcal{S}^* = \{a_i\}_{i=1..m}$ where $|\mathcal{S}^*| = m$. $\mathcal{A}$ sends $\mathcal{S}^*$ to $\mathcal{C}$.

SETUP – the challenger $\mathcal{C}$ runs the $\mathsf{setup}(\xi)$ algorithm, sends the public parameters $\mathtt{pp}$ to the adversary $\mathcal{A}$ and keeps secret the master key $\mathtt{msk}$.

QUERY PHASE 1 – the adversary $\mathcal{A}$ queries, for each session i, an access policy $\Psi_{\mathcal{A},i}$. The challenger $\mathcal{C}$ answers by running the $\mathsf{keygen}(\mathtt{pp},\mathtt{msk},\Psi_{\mathcal{A},i})$ algorithm and sends the resulting secret key $sk_{\mathcal{A},i}$ to the adversary $\mathcal{A}$. Note that the access policy $\Psi_{\mathcal{A},i}$ does not satisfy the encryption attribute set $\mathcal{S}^*$.

CHALLENGE PHASE – during the challenge phase, $\mathcal{A}$ picks two equal length cleartexts $M_0{}^*$ and $M_1{}^*$ as well as an attribute set $\mathbb{U}^*$ with $|\mathbb{U}^*| = t$ $\mathbb{U}^* \cap \mathcal{S}^* = \emptyset$ if $\mathsf{ind} = \mathsf{add}$ or $\mathbb{U}^* \subset \mathcal{S}^*$ if $\mathsf{ind} = \mathsf{revoke}$. The challenger $\mathcal{C}$ chooses a random bit b from $\{0,1\}$ with $\mathcal{S}'^* = \mathcal{S}^* \setminus \mathcal{U}^*$ for $\mathsf{ind} = \mathsf{add}$ or $\mathcal{S}'^* = \mathcal{S}^* \cap \mathcal{U}^*$ for $\mathsf{ind} = \mathsf{revoke}$ and computes the challenge encrypted message $CT_b{}^* = \mathsf{encrypt}(\mathtt{pp},\mathcal{S}^*,M_b{}^*)$. It gives $CT_b{}^*$ to the adversary if $\mathcal{U} = \emptyset$, otherwise $CT_b'^* = \mathsf{update}(\mathtt{pp},CT_b{}^*,\mathsf{ind},\mathcal{U}^*)$.

QUERY PHASE 2 – in this phase, the adversary $\mathcal{A}$ can query a polynomially bounded number of queries as in QUERY PHASE 1, except that $\mathcal{A}$ cannot query secret keys related to a set of attributes $\mathcal{S}^*$.

GUESS – $\mathcal{A}$ tries to guess which message M_i, where $i \in \{0,1\}$ corresponds to the enciphered data $CT_b{}^*$. Thus, $\mathcal{A}$ outputs a bit b' of b and wins the game if $b = b'$. The advantage of the adversary $\mathcal{A}$ in the above game is defined as $Adv_{\mathcal{A}}[Exp^{Conf}(1^{\xi})] = |Pr[b = b'] - \frac{1}{2}|$.

5 Mathematical Background

In this section, we first introduce the access structure in section 5.1. Then, in section 5.2, we present the bilinear maps. Finally, we introduce some security assumptions.

5.1 Access Policies

Access policies can be represented as a boolean functions of attributes or a Linear Secret Sharing Scheme (LSSS) matrix.

Access Structure – Let $\mathcal{P} = \{\mathcal{P}_1,\cdots,\mathcal{P}_n\}$ be a set of parties. A collection $\mathtt{A} \subseteq 2^{\{\mathcal{P}_1,\cdots,\mathcal{P}_n\}}$ is monotone if $\forall B,C$ if $B \in \mathtt{A}$ and $B \subseteq C$ then $C \in \mathtt{A}$.
An access structure is a collection $\mathtt{A}$ of non-empty subsets of $\{\mathcal{P}_1,\cdots,\mathcal{P}_n\}$, such as $\mathtt{A} \subseteq 2^{\{\mathcal{P}_1,\cdots,\mathcal{P}_n\}} \setminus \emptyset$. Note that any access structure can be converted into a boolean function. Boolean functions can be defined as an access tree, where the leaves present the attributes while the intermediate and the root nodes are the logical operators AND ($\wedge$) and OR ($\vee$).

Linear Secret Sharing Schemes (LSSS) – Let $\mathcal{P}$ be a set of parties, A be $l \times n$ matrix, and $\rho : \{1,2,\cdots,l\} \rightarrow \mathcal{P}$ be a function that maps a row to a party for labeling. A secret sharing scheme for access structure Ψ over a set of parties $\mathcal{P}$ is a linear secret sharing scheme (LSSS) in $\mathbb{Z}_p$ and is represented by (A,ρ) if it consists of two efficient algorithms:

- $\mathsf{Share}((A,\rho),s)$: The share algorithm takes as input $s \in \mathbb{Z}_p$ which is to be shared. It randomly chooses $\beta_1,\cdots,\beta_n \in \mathbb{Z}_p$, and defines $\beta = (\beta_1 = s,\beta_2,\cdots,\beta_k)^T$. It outputs $M.\beta$ as the vectors of l shares. The share $\lambda_i =< A_i,\beta^T >$ belongs to party $\rho(i)$, where A_i is the $i-th$ of A.

- $\mathsf{Recon}((A,\rho),\mathcal{S})$: The reconstruction algorithm takes as input an access set $\mathcal{S} \in A$. Let $I = \{i|\rho(i) \in \mathcal{S}\}$. It outputs a set of constants $\{\mu_i\}_{i\in I}$ such that $\sum_{i\in I} \mu_i \cdot \lambda_i = \beta_1 = s$.

5.2 Bilinear Maps

Let $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ be three multiplicative groups of a finite field having the same order p. An admissible asymmetric pairing function $\hat{e}$ from $\mathbb{G}_1 \times \mathbb{G}_2$ in $\mathbb{G}_T$ has to be bilinear, non degenerate and efficiently computable.

5.3 The General Diffie-Hellman Exponent Assumption

In our PU-ABE construction, we make use of the generalisation of the Diffie-Hellman exponent assumption, formally defined by Boneh et al. in (Boneh et al., 2005). The authors have introduced a class of assumptions that appeared with the use of pairing-based schemes namely Decisional DiffieHellman assumption (DDH), Bilinear Diffie-Hellman (BDH), and q-Bilinear Diffie Hellman Exponent (qBDHE) assumptions, detailed hereafter.

Let $B = (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}, \hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G})$ be a bilinear map group such that $\mathbb{G}_1 = \mathbb{G}_2 = \mathbb{G}$. Let g_0 be a generator of $\mathbb{G}$ and set $g = \hat{e}(g_0, g_0) \in \mathbb{G}$. Let s and n be positive integers and $P, Q \in \mathbb{F}_p[X_1, \cdots, X_n]^s$ be two s-tuples of n-variate polynomials over $\mathbb{F}_p$ where $P = (p_1, \cdots, p_s)$ and $Q = (q_1, \cdots, q_s)$ and $p_1 = q_1 = 1$. For any function $h : \mathbb{F}_p \to \Omega$ and any vector $(x_1, \cdots, x_n) \in \mathbb{F}_p^n$, $h(P(x_1, \cdots, x_n))$ stands for $(h(p_1(x_1, \cdots, x_n)), \cdots, h(p_s(x_1, \cdots, x_n))) \in \Omega^s$ and $h(Q(x_1, \cdots, x_n))$ stands for $(h(q_1(x_1, \cdots, x_n)), \cdots, h(q_s(x_1, \cdots, x_n))) \in \Omega^s$. Let $f \in \mathbb{F}_p[X_1, \cdots, X_n]$, it is said that f depends on (P, Q), which we denote by $f \in P, Q$, when there is a linear decomposition

$$f = \sum_{1\le i,j\le s} a_{i,j} \cdot p_i \cdot p_j + \sum_{1\le i\le s} b_i \cdot q_I, \quad a_{i,j}, b_i \in \mathbb{Z}_p$$

Let P, Q be as above and $f \in \mathbb{F}_p[X_1, \cdots, X_n]$. The (P, Q, f)-General Diffie- Hellman Exponent problems are defined as follows.

Definition 1: (P, Q, f)***-GDHE***. Given a tuple $H(x_1, \cdots, x_n) = (g_0^{P(x_1,\cdots,x_n)}, g^{Q(x_1,\cdots,x_n)}) \in \mathbb{G}_1^s \times \mathbb{G}^s$, compute $g^{f(x_1,\cdots,x_n)}$.

Definition 2: (P, Q, f)***-GDDHE***. Given $H(x_1, \cdots, x_n) \in \mathbb{G}_1^s \times \mathbb{G}^s$, compute $g^{f(x_1,\cdots,x_n)}$ as above, decide whether $T = g^{f(x_1,\cdots,x_n)}$.

We refer to (Boneh et al., 2005) for a proof that (P, Q, f)***-GDHE*** and (P, Q, f)***-GDDHE*** have generic security when $f \notin < P, Q >$.

6 A NOVEL KEY POLICY ATTRIBUTE BASED ENCRYPTION SUPPORTING POLICY UPDATE

In this section, we first give an overview of our proposed constant size PU-ABE scheme (subsection 6.1). Then, we detail the scheme construction.

6.1 Overview

In this paper, we develop a novel key-policy attribute based encryption scheme that supports access policy update. Our contribution extends Wang et al.'s key policy attribute based encryption scheme (Wang and Luo, 2012) to support adding new attributes or revoking others from the encryption access policy. Indeed, the cloud server is able to change a ciphertext encrypted with respect to an access policy $\mathcal{S}$ to a ciphertext encrypted with respect to a new access policy $\mathcal{S}'$ such that $\mathcal{S}'$ corresponds to $\mathcal{S} \cup \mathcal{U}$ in case of attributes' addition and $\mathcal{S} \setminus \mathcal{U}$ in case of attributes' revocation. This update does not need any re-encryption key neither the data owner to be online. It is only based on some extra ciphertext components as detailed in Section 6.2) which are used by the cloud server in the update algorithm. Indeed, while encrypting data contents, $\mathcal{E}$ generates additional elements in the ciphertext which point out attributes that can be added or removed later. When an access policy update is required, the cloud server uses the ciphertext's components to generate a new ciphertext suitable for the new access policy then forwards it to the decrypting entity.

6.2 Concrete Construction

Our PU-ABE construction, supporting both attributes' addition and revocation, is based on five algorithms defined as follows:

- setup – given the security parameter ξ, the attribute authority chooses three cyclic groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ of prime order p and defines a bilinear pairing $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$. It also randomly selects two generators $g \in \mathbb{G}_1$ and $h \in \mathbb{G}_2$ as well as a secret random value $\alpha \in \mathbb{Z}_p^*$. In addition, the attribute authority sets $v = \hat{e}(g, h)$, $\{h^{\alpha^i}\}_{i=1\cdots k}$ and $\{u_i = g^{\alpha^i}\}_{i=1\cdots k}$ where $k = |\mathbb{U}|$ is the cardinal of

the attributes universes $\mathbb{U}$. Finally, it chooses a cryptographic hash function $\mathcal{H}: \{0,1\}^* \Rightarrow \mathbb{Z}_p^*$ and outputs the public parameters pp as follows:

$$\begin{aligned} \texttt{pp} &= (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, v = \hat{e}(g,h), h, \{h^{\alpha^i}\}_{i=1\cdots k}, \\ &\quad \{u_i = g^{\alpha^i}\}_{i=1\cdots k}) \end{aligned}$$

The master secret key is defined as $msk = (g, \alpha)$.

- encrypt – let $\mathcal{S}$ be the set of the encryption attributes $\mathcal{S} = \{a_i\}_{i=1}^m$. This algorithm, executed by the encrypting entity $\mathcal{E}$ takes an extra input which is a maximum revocation number $r \leq m$. The data owner chooses $s \in \mathbb{Z}_p^*$ and computes the ciphertext CT as follows:

$$\begin{cases} E_0 = h^{s \cdot \prod_{a_i \in \mathcal{S}} (\alpha + \mathcal{H}(a_i))} \\ E_1 = E_0^{\alpha}, \cdots, E_{k-m} = E_{k-m-1}^{\alpha} \\ C_1 = u_1^{-s} =, \cdots, C_{r+1} = u_{r+1}^{-s} \\ C_M = M \cdot \hat{e}(g,h)^s \end{cases}$$

- update – the update algorithm first checks the operation indicator ind. Then, if ind = add, it proceeds as (i), otherwise if ind = revoke it executes (ii):

(i) – given a ciphertext CT encrypted w.r.t. a set of attributes $\mathcal{S}$ and $\mathcal{U} = \{a'_1, \cdots, a'_t\}$ a new set of attributes where $\mathcal{U} \cap \mathcal{S} = \emptyset$, the server has to add elements of $\mathcal{U}$ to the set of encrypting attributes $\mathcal{S}$ of the ciphertext CT. To do so, it proceeds as follows:
Let $F(x)$ be the polynomial in x defined as $F(x) = \prod_{a_i \in \mathcal{U}} (x + \mathcal{H}(a_i)) = f_t x^t + f_{t-1} x^{t-1} + \cdots + f_0$
Then, the algorithm computes $E'_0 = E_0^{F(\alpha)} = \prod_{i=0}^t E_i^{f_i}$. The new ciphertext is then defined as $CT' = (E'_0, C_1, C_M)$ w.r.t. $\mathcal{S}'$, the new set of encrypting attributes defined as $\mathcal{S}' = \mathcal{S} \cup \mathcal{U}$.

(ii) – given a ciphertext CT encrypted w.r.t. a set of attributes $\mathcal{S}$ and a revocation attribute set $\mathcal{U} = \{a'_1, \cdots, a'_t\} \subseteq \mathcal{S}$ where $t \leq r$, the server updates the ciphertext CT as follows:
Let $F(x)$ be the polynomial in x as $F(x) = \frac{1}{\prod_{a_i \in U} \mathcal{H}(a_i)} \prod_{a_i \in \mathcal{U}} (x + \mathcal{H}(a_i)) = f_t x^t + f_{t-1} x^{t-1} + \cdots + f_0$
Then, the algorithm computes CT' as follows:

$$\begin{cases} E'_0 = E_0^{\frac{1}{\prod_{a_i \in \mathcal{U}} \mathcal{H}(a_i)}} = h^{s \cdot \prod_{a_i \in \mathcal{S} \setminus \mathcal{U}} (\alpha + \mathcal{H}(a_i)) F(\alpha)} \\ C'_1 = \prod_{t=1}^{t+1} C_i^{f_{i-1}} = g^{-\alpha s \sum_{i=1}^{t+1} \alpha^{i-1} f_{i-1}} = u_1^{-sF(\alpha)} \\ C'_M = C_M \hat{e}(\prod_{i=1}^t C_i^{-f_i}, h) = M \hat{e}(g,h)^{s \sum_{i=0}^t f_i \alpha^i} \\ = M v^{sF(\alpha)} \end{cases}$$

- keygen – it computes the private key associated to an access structure Ψ w.r.t. an LSSS scheme (A, ρ) such that A is the corresponding $l \times n$ matrix. First, the keygen algorithm generates shares of 1 relying on the LSSS schema w.r.t. (A, ρ), as detailed in section 5. Namely, it chooses a column vector $\beta = (\beta_1, \beta_2, \cdots, \beta_n)^T$, while $\beta_1 = s = 1$ and $\beta_2, \cdots, \beta_n \in \mathbb{Z}_p$. Then for each $i = 1$ to l, it calculates $\lambda_i = < A_i, \beta^T >$, and sets sk as follows:

$$\begin{aligned} sk &= \{D_i, (K_{i,j})_{j=0}^n\}_{i=0}^l \\ &= \{g^{\frac{\lambda_i}{\alpha + \mathcal{H}(\rho(i))}}, (h^{\lambda_i \alpha^j})_{j=0}^n\}_{i=0}^l \end{aligned}$$

- decrypt – the cipherext CT' is encrypted under the set of attributes $\mathcal{S}' = \mathcal{S} \cup \mathcal{U} = \{a_{i=1}^{t+m}\}$. The decrypting entity $\mathcal{D}$ having a secret key $sk_{(A,\rho)}$ first sets $I = i | \rho(i) \in \mathcal{S}'$, and calculates the reconstruction of constants $\mu_{i \in I} = \mathsf{Recon}((A, \rho), \mathcal{S})$. The decryption key corresponding to the LSSS scheme w.r.t. (A, ρ) is parsed as $sk = \{D_i, (K_{i,j})_{j=0}^n\}_{i=0}^l$. Then, $\mathcal{D}$ computes the polynomial on the variable α with degree $m + t - 1$ as follows:

$$P_{i,A(\alpha)} = \frac{\lambda_i}{\alpha} \left(\prod_{j=1, j \neq i} (\alpha + \mathcal{H}(a_j)) - \prod_{j=1, j \neq i} \mathcal{H}(a_j) \right)$$

$\mathcal{D}$ calculates $h^{P_{i,A(\alpha)}}$ according to his secret key component $(K_{i,j})_{j=0}^n$. Afterwards, she computes Y_i which can be retrieved based on two cases w.r.t. the ind operator value, such that:

– **Case 1:** if attributes have been added to the access policy:

$$\begin{aligned} Y_i &= (\hat{e}(C_1, h^{P_{i,A(\alpha)}}) \cdot \hat{e}(D_i, C_2))^{\frac{1}{\prod_{j=1, j \neq i} \mathcal{H}(a_i))}} \\ &= \hat{e}(g,h)^{s \lambda_i} \end{aligned}$$

Finally, $\mathcal{D}$ computes $Y = \prod_{i \in I} Y_i^{\mu_i} = \hat{e}(g,h)^s$ and retrieves $M = C_M / Y$. Note that if no changes have been made to the access policy and the corresponding ciphertext, the decryption process follows **Case 1**.

– **Case 2:** if attributes have been revoked from the access policy:

$$\begin{aligned} Y_i &= (\hat{e}(C'_1, h^{P_{i,A(\alpha)}}) \cdot e(D_i, E'_0))^{\frac{1}{\prod_{j=1, j \neq i} \mathcal{H}(a_i))}} \\ &= \hat{e}(g,h)^{sF(\alpha) \lambda_i} \end{aligned}$$

Finally, $\mathcal{D}$ computes $Y = \prod_{i \in I} Y_i^{\mu_i} = \hat{e}(g,h)^s$ and retrieves $M = C'_M / Y$.

7 Security analysis

In this section, we first prove the correctness of our PU-ABE construction, with respect to the data decryption algorithms proposed in the previous construction, in section 7.1. Then, we prove the security of our proposal, with respect to the indistinguishability property in Section 7.2.

7.1 Correctness

In the following, we prove the correctness of the PU-ABE proposed construction w.r.t. the attributes' addition and revocation. A decrypting entity $\mathcal{D}$ who possesses a set of attributes expressed with respect to an access structure Ψ, satisfying $\mathcal{S}'$ first sets $P_{i,A(\alpha)} = \frac{\lambda_i}{\alpha}(\prod_{j=1,j\neq i}(\alpha+\mathcal{H}(a_j)) - \prod_{j=1,j\neq i}\mathcal{H}(a_j)))$. Then, $\mathcal{D}$ uses his secret keys to compute Y_i with respect to the two following cases:

- **Case 1:** if ind = add and $\mathcal{S}' = \mathcal{S} \cup \mathcal{U}$, then Y_i is computed as:

$$\begin{aligned}
Y_i &= [\hat{e}(C_1', h^{P_{i,A(\alpha)}}) \cdot \hat{e}(D_i, E_0')]^{\frac{1}{\prod_{j=1,j\neq i}\mathcal{H}(a_j))}} \\
&= [\hat{e}(g^{-\alpha s}, h^{P_{i,A(\alpha)}}) \cdot \hat{e}(g^{\frac{\lambda_i}{\alpha+\mathcal{H}(\rho(i))}}, E_0^{F(\alpha)})]^{\frac{1}{\prod_{j=1,j\neq i}\mathcal{H}(a_j))}} \\
&= [\hat{e}(g^{-\alpha s}, h^{P_{i,A(\alpha)}}) \\
&\quad \cdot \hat{e}(g^{\frac{\lambda_i}{\alpha+\mathcal{H}(\rho(i))}}, h^{s\cdot\prod_{a_j\in\mathbb{A}}(\alpha+\mathcal{H}(a_j))F(\alpha)})]^{\frac{1}{\prod_{j=1,j\neq i}\mathcal{H}(a_j))}} \\
&= [\hat{e}(g,h)^{-\alpha s P_{i,A(\alpha)}} \\
&\quad \cdot \hat{e}(g,h)^{s\lambda_i F(\alpha)\cdot\prod_{j=1,j\neq i}(\alpha+\mathcal{H}(a_j))})]^{\frac{1}{\prod_{j=1,j\neq i}\mathcal{H}(a_j))}} \\
&= [\hat{e}(g,h)^{-s\lambda_i(\prod_{j=1,j\neq i}(\alpha+\mathcal{H}(a_j))} \hat{e}(g,h)^{s\lambda_i\prod_{j=1,j\neq i}\mathcal{H}(a_j)} \\
&\quad \cdot \hat{e}(g,h)^{s\lambda_i F(\alpha)\cdot\prod_{j=1,j\neq i}(\alpha+\mathcal{H}(a_j))}]^{\frac{1}{\prod_{j=1,j\neq i}\mathcal{H}(a_j)}} \\
&= [\hat{e}(g,h)^{s\lambda_i\prod_{j=1,j\neq i}\mathcal{H}(a_j)}]^{\frac{1}{\prod_{j=1,j\neq i}\mathcal{H}(a_j)}} \\
&= \hat{e}(g,h)^{s\lambda_i}
\end{aligned}$$

Afterwards, the decrypting entity $\mathcal{D}$ computes:

$$\begin{aligned}
Y &= \prod_{i\in I} Y_i^{\mu_i} \\
&= \prod_{i\in I} \hat{e}(g,h)^{s\lambda_i\mu_i} \\
&= \hat{e}(g,h)^{s\sum_{i\in I}\lambda_i\mu_i} \\
&= \hat{e}(g,h)^{s}
\end{aligned}$$

Recall that the constants $mu_{i\in I}$ are the reconstruction of the LSSS matrix $\mu_{i\in I} = \mathsf{Recon}((A,\rho),\mathbb{A})$. Therefore, the user retrieves Y using the following equation

$$< \lambda, \mu >= \sum_{i\in I}\lambda_i\mu_i = \sum_{i\in I}\beta_1 = \sum_{i\in I} 1 = 1$$

Finally, $\mathcal{D}$ retrieves M such as:

$$M = \frac{C_M}{Y} = \frac{C_M}{\hat{e}(g,h)^s} = \frac{M\hat{e}(g,h)^s}{\hat{e}(g,h)^s}$$

- **Case 2:** if ind = revoke and $\mathcal{S}' = \mathcal{S} \setminus \mathcal{U}$, Y_i is computed as follows:

$$\begin{aligned}
Y_i &= [\hat{e}(C_1', h^{P_{i,A(\alpha)}}) \cdot \hat{e}(D_i, E_0')]^{\frac{1}{\prod_{j=1,j\neq i}\mathcal{H}(a_j))}} \\
&= [\hat{e}(u_1^{-sF(\alpha)}, h^{P_{i,A(\alpha)}}) \cdot \hat{e}(g^{\frac{\lambda_i}{\alpha+\mathcal{H}(\rho(j))}}, \\
&\quad h^{s\cdot\prod_{a_j\in\mathbb{A}\setminus\mathbb{U}}(\alpha+\mathcal{H}(a_j))F(\alpha)})]^{\frac{1}{\prod_{j=1,j\neq i}\mathcal{H}(a_j))}} \\
&= [\hat{e}(g,h)^{-s\lambda_i F(\alpha)(\prod_{j=1,j\neq i}(\alpha+\mathcal{H}(a_i))-\prod_{j=1,j\neq i}\mathcal{H}(a_i)))} \cdot \\
&\quad \hat{e}(g,h)^{s\lambda_i\cdot\prod_{j=1,j\neq i}(\alpha+\mathcal{H}(a_j))F(\alpha)})]^{\frac{1}{\prod_{j=1,j\neq i}\mathcal{H}(a_j))}} \\
&= [\hat{e}(g,h)^{-s\lambda_i F(\alpha)(\prod_{j=1,j\neq i}(\alpha+\mathcal{H}(a_i))} \\
&\quad \cdot \hat{e}(g,h)^{s\lambda_i F(\alpha)\prod_{j=1,j\neq i}\mathcal{H}(a_i)))} \\
&\quad \cdot \hat{e}(g,h)^{s\lambda_i F(\alpha)\cdot\prod_{j=1,j\neq i}(\alpha+\mathcal{H}(a_j)))}]^{\frac{1}{\prod_{j=1,j\neq i}\mathcal{H}(a_j))}} \\
&= \hat{e}(g,h)^{s\lambda_i F(\alpha))\frac{\prod_{j=1,j\neq i}\mathcal{H}(a_i)}{\prod_{j=1,j\neq i}\mathcal{H}(a_j)}} \\
&= \hat{e}(g,h)^{sF(\alpha)\lambda_i}
\end{aligned}$$

Afterwards, the decrypting entity computes:

$$\begin{aligned}
Y &= \prod_{i\in I} Y_i^{\mu_i} \\
&= \prod_{i\in I} \hat{e}(g,h)^{sF(\alpha)\lambda_i\mu_i} \\
&= \hat{e}(g,h)^{sF(\alpha)\sum_{i\in I}\lambda_i\mu_i} \\
&= \hat{e}(g,h)^{sF(\alpha)}
\end{aligned}$$

Finally, $\mathcal{D}$ retrieves M such as:

$$M = \frac{C_M'}{Y} = \frac{C_M'}{\hat{e}(g,h)^{sF(\alpha)}} = \frac{M\hat{e}(g,h)^{sF(\alpha)}}{\hat{e}(g,h)^{sF(\alpha)}}$$

7.2 Confidentiality

In the following proof, we prove that our PU-ABE scheme is CPA-Secure against non-adaptive Chosen Ciphertext Attacks with respect to Theorem 1.

Theorem 1. *For any adversary $\mathcal{A}$, against CPA-Secure against non-adaptive chosen ciphertext, our PU-ABE scheme is indistinguishable according to Definition 4.2 with respect to the hardness of the General Diffie-Hellman Exponent (GDHE) assumption (Definition 5.3)*

Proof. To decrypt a ciphertext CT' associated with an updated access policy $\mathcal{S}'$, $\mathcal{A}$ must recover $\hat{e}(g,h)^s$, in case of attributes' addition and $\hat{e}(g,h)^{s\sum_{i=0}^{t} f_i\alpha^i}$, in case

of attributes' revocation, where the secret sharing key s is embedded in the ciphertext. For this purpose, $\mathcal{A}$ has to retrieve the corresponding $\tilde{C}_M$ and the related private key.

To prove that our scheme is secure against selective non-adaptive chosen ciphertext attacks, we first consider that $\mathcal{A}$ is running the Exp^{conf} experiment with an entity $\mathcal{B}$. This latter is running the $Exp_{\mathcal{B}}$. Wang et al. security game (Wang and Luo, 2012), with $\mathcal{C}$. The objective of this proof is to show that the advantage of $\mathcal{A}$ to win the $G^{S-CPA}(1^{\kappa})$ security game is equivalent to the advantage of $Exp_{\mathcal{B}}$ to win the Wang et al. security game (Wang and Luo, 2012). Hereafter, $\mathcal{A}$ and $\mathcal{B}$ proceed as follows:

INITIALISATION – in this phase, the adversary $\mathcal{A}$ gives the algorithm $\mathcal{C}$ a challenge set of attributes $\mathcal{S}^*$.

SETUP – the challenger $\mathcal{C}$ runs the $\mathsf{setup}(\xi)$ algorithm, sends the public parameters $\mathsf{pp} = (\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, v = \hat{e}(g,h), g^{\alpha}, h, h^{\alpha}, \cdots, h^{\alpha^k}, \{u_i = g^{\alpha^i}\}_{i=1\cdots k})$ to $\mathcal{B}$ and keeps secret $\mathtt{msk}$. Consequently, $\mathcal{B}$ sends pp to $\mathcal{A}$.

QUERY PHASE 1 – $\mathcal{B}$ sets an empty table T and repeatedly queries an access policy $\Psi_{\mathcal{A},i}$, for each session i. That is, $\mathcal{B}$ uses $\mathcal{C}$ to derive and send the queried secret keys to $\mathcal{A}$. The challenger $\mathcal{C}$ answers by running the $\mathsf{keygen}(\mathsf{pp}, \mathtt{msk}, \Psi_{\mathcal{A},i})$ algorithm. The challenger $\mathcal{C}$ generates shares of 1 relying on the LSSS schema w.r.t. (A, ρ). It chooses a column vector $\beta = (\beta_1, \beta_2, \cdots, \beta_n)^T$, while $\beta_1 = s = 1$ and $\beta_2, \cdots, \beta_n \in \mathbb{Z}_p$. Then for each $i = 1$ to l, it calculates $\lambda_i =< A_i, \beta^T >$, and sets sk as $sk_{\mathcal{A},i} = \{g^{\frac{\lambda_i}{\alpha+\mathcal{H}(\rho(i))}}, (h^{\lambda_i \alpha^j})_{j=0}^{n}\}_{i=0}^{l}$.

Note that the access policy $\Psi_{\mathcal{A},i}$ does not satisfy the encryption attribute set $\mathcal{S}^*$. The private keys $sk_{\mathcal{A},i} = \{g^{\frac{\lambda_i}{\alpha+\mathcal{H}(\rho(i))}}, (h^{\lambda_i \alpha^j})_{j=0}^{n}\}_{i=0}^{l}$ are returned to $\mathcal{B}$. Subsequently, $\mathcal{B}$ sets a new entry with the private key and returns $sk_{\mathcal{A},i} = \{g^{\frac{\lambda_i}{\alpha+\mathcal{H}(\rho(i))}}, (h^{\lambda_i \alpha^j})_{j=0}^{n}\}_{i=0}^{l}$ to $\mathcal{A}$.

CHALLENGE PHASE – during the challenge phase, $\mathcal{A}$ picks two equal length cleartexts $M_0{}^*$ and $M_1{}^*$ as well as an attribute set $\mathbb{U}^*$ with $|\mathbb{U}^*| = t$ $\mathbb{U}^* \cap \mathcal{S}^* = \emptyset$ if $\mathsf{ind} = \mathsf{add}$ or $\mathbb{U}^* \subset \mathcal{S}^*$ if $\mathsf{ind} = \mathsf{revoke}$. Subsequently, $\mathcal{B}$ selects $\mathcal{S}_{\mathcal{B}}$ such that $\mathcal{S}_{\mathcal{B}} \subseteq \mathcal{S}'^*$, such as $\mathcal{S}'^* = \mathcal{S}^* \setminus \mathcal{U}^*$ for $\mathsf{ind} = \mathsf{add}$ or $\mathcal{S}'^* = \mathcal{S}^* \cap \mathcal{U}^*$ for $\mathsf{ind} = \mathsf{revoke}$.
Afterwards, $\mathcal{B}$ sends the access structure $\mathcal{S}_{\mathcal{B}}$ and the two equal length messages M_0 and M_1, defined by $\mathcal{A}$ to the challenger $\mathcal{C}$. The challenger $\mathcal{C}$ chooses a random bit b from $\{0,1\}$ with $\mathcal{S}'_{\mathcal{B}} = \mathcal{S}_{\mathcal{B}} \setminus \mathcal{U}^*$ for $\mathsf{ind} = \mathsf{add}$ or $\mathcal{S}'_{\mathcal{B}} = \mathcal{S}_{\mathcal{B}} \cap \mathcal{U}^*$ for $\mathsf{ind} = \mathsf{revoke}$ and computes the challenge encrypted message $CT_b{}^* = \mathsf{encrypt}(\mathsf{pp}, \mathcal{S}'_{\mathcal{B}}, M_b{}^*)$.

$$\begin{cases} E_0 = h^{s \cdot \prod_{a_i \in \mathcal{S}'_{\mathcal{B}}} (\alpha + \mathcal{H}(a_i))} \\ E_1 = E_0^{\alpha}, \cdots, E_{p-m} = E_{p-m-1} \\ C_1 = u_1^{-s} =, \cdots, C_{r+1} = u_{r+1}^{-s} \\ C_M = M \cdot v^s = M \cdot \hat{e}(g,h)^s \end{cases}$$

The challenger $\mathcal{C}$ gives $CT_b{}^*$ to the adversary if $\mathcal{U} = \emptyset$, otherwise $CT'^*_b = \mathsf{update}(\mathsf{pp}, CT_b{}^*, \mathsf{ind}, \mathcal{U}^*)$.

QUERY PHASE 2 – in this phase, the adversary $\mathcal{A}$ can query a polynomially bounded number of queries as in QUERY PHASE 1, except that $\mathcal{A}$ cannot query secret keys related to a set of attributes $\mathcal{S}^*$.

Hereafter, two cases are considered w.r.t. the ind operator value, randomly selected by $\mathcal{C}$ in order to encrypt the challenging message such that:

- **Case 1** – the first case corresponds to attributes' addition, such that $\mathcal{C}$ sets $\mathcal{S}'_{\mathcal{B}} = \mathcal{S}_{\mathcal{B}} \cup \mathcal{U}^*$ and outputs an encrypted message CT'_b, as defined in section 6.2. In this case, we first show that how a challenge ciphertext should be produced. In fact, given a ciphertext CT encrypted w.r.t. a set of attributes $\mathcal{S}_{\mathcal{B}}$ and $\mathcal{U}^* = \{a'_1, \cdots, a'_t\}$ a new set of attributes where $\mathcal{U} \cap \mathcal{S} = \emptyset$, $\mathcal{C}$ has to add elements of $\mathcal{U}^*$ to the set of encrypting attributes $\mathcal{S}_{\mathcal{B}}$ of the ciphertext CT'_b. To do so, it proceeds as follows: Let $F(x)$ be the polynomial in x defined as $F(x) = \prod_{a_i \in \mathcal{U}^*}(x + \mathcal{H}(a'_i)) = f_t x^t + f_{t-1} x^{t-1} + \cdots + f_0$ Then, the algorithm computes $E'^*{}_0 = E^{*F(\alpha)}_0 = \prod_{i=0}^{t} E_i^{f_i}$. The new ciphertext is then defined as $CT'_b = (E'^*{}_0, C_1, C_M)$ w.r.t. $\mathcal{S}'$, the new set of encrypting attributes defined as $\mathcal{S}'_{\mathcal{B}} = \mathcal{S}_{\mathcal{B}} \cup \mathcal{U}^*$.
- **Case 2** – the second case corresponds to attributes' revocation, such that $\mathcal{C}$ defines $\mathcal{S}'_{\mathcal{B}} = \mathcal{S}_{\mathcal{B}} \setminus \mathcal{U}^*$ and outputs an encrypted message CT'_b, as detailed in section 6.2. That is, given a ciphertext CT encrypted w.r.t. a set of attributes $\mathcal{S}'_{\mathcal{B}}$ and a revocation attribute set $\mathcal{U}^* = \{a'_1, \cdots, a'_t\} \subseteq \mathcal{S}'_{\mathcal{B}}$ where $t \leq r$, the server updates the ciphertext CT'_b as follows: Let $F(x)$ be the polynomial in x as $F(x) = \frac{1}{\prod_{a'_i \in \mathcal{U}^*} \mathcal{H}(a'_i)} \prod_{a'_i \in \mathcal{U}} (x + \mathcal{H}(a'_i)) = f_t x^t + f_{t-1} x^{t-1} + \cdots + f_0$. Similarly, the new ciphertext is then defined as $CT'_b = (E'^*{}_0, C'^*{}_1, C'^*{}_M)$ w.r.t. $\mathcal{S}'$, the new set of encrypting attributes defined as $\mathcal{S}'_{\mathcal{B}} = \mathcal{S}_{\mathcal{B}} \setminus \mathcal{U}^*$.

Without loss of generality, the distribution of the received challenge ciphertext does not depend on the attributes' addition and revocation. More precisely,

the distribution of the challenge enciphered message is quite similar in both cases. Thus, the resistance of PU-ABE scheme against CPA, follows (Wang and Luo, 2012) construction, w.r.t. to $\mathcal{B}$, that is proven secure under the GDDHE assumption. Thus, the view of $\mathcal{B}$ is indistinguishable from the view of $\mathcal{A}$, considering a randomly selected enciphered message w.r.t. $\mathcal{S}'_{\mathcal{B}}$ referring to the updated access policy.

As such, we prove that our PU-ABE construction is secure against selective non-adaptive chosen ciphertexts attacks in the standard model, under the GDDHE assumption, with respect to Exp^{conf} security experiment. □

8 PERFORMANCES ANALYSIS

In this section, we present the computation and the storage complexities of our proposed PU-ABE scheme. In our analysis, we are interested in the computations performed to execute the encrypt, update and the decrypt algorithms as well as the size of the generated encrypted message and the size of the secret keys as introduced in Table 1.

8.1 Storage Complexities

Emura et al. (Emura et al., 2009) introduced a KP-ABE scheme requiring only 2 keys per user independent of the users' attributes. In addition, this scheme generates a ciphertext composed only of 3 elements.
Herranz et al. (Herranz et al., 2010) have proposed the first CP-ABE scheme generating a ciphertext whose size does not depend on the number of attributes used in the threshold access policy. In this scheme, the decrypting entity needs $k+n$ secret keys' where k is the cardinal of the attributes universes and n is the number of the users' attributes.
In (Ge et al., 2012), the authors proposes a CP-ABE scheme with constant ciphertext size. However, this schemes requires the use of $3k-2+n$ secret keys.
Similarly, the authors in (Wang and Luo, 2012) proposed a KP-ABE scheme which produces only 3 elements in the ciphertext. The users' secret keys are equal to $k(n+1)$. This size of secret keys is due to the use of LSSS monotone access policies which makes the scheme more expressive than the aforementioned schemes.

Although, the above schemes ensure low storage and communication costs, they do not support access policy updates. Indeed, if the access rights change with the addition or the revocation of some attributes, outsourceddata need to be re-encrypted.

Jiang et al. (Jiang et al., 2017), have proposed a threshold CP-ABE scheme supporting access policy update. The authors proposed two different construction. The first construction ensures the addition of attributes to the access policy. This incurs the generation of a ciphertext whose size is equal to $3+k-m$ to be forwarded to the cloud server however the final user only receives 3 elements of the ciphertext no matter how many attributes are used in the access policy. The second construction provides the ability to revoke attributes from the access policy. Therefore, the generated access policy depends on the maximum number of attributes in an attribute revocation list. Like the first construction, the user only needs three elements to decrypt data. Both the proposed construction require $n+1$ secret keys for every user.

In our PU-ABE scheme, we apply a compact policy update technique to ensure adding and/or removing attributes from access policies in KP-ABE schemes. Therefore, the proposed construction generates a ciphertext size equal to $3+k+r-m$. Users receives a constant ciphertext size independent from the number of attributes involved in the access policy and from the applied update procedures. PU-ABE relies in using monotone access policies, then the users secret keys are equal to $n+k$ elements. Therefore, the proposed PU-ABE scheme ensures expressiveness and policy updates while introducing comparative storage with similar ABE schemes.

8.2 Computation Complexities

The proposed schemes in (Emura et al., 2009), (Herranz et al., 2010) and (Wang and Luo, 2012) introduce an encryption algorithm which requires two exponentiations in $\mathbb{G}_1$ and one exponentiation in $\mathbb{G}$. Ge et al.'s scheme (Ge et al., 2012) introduces an encryption algorithm requiring 5 exponentiations in $\mathbb{G}_1$ and one exponentiation in $\mathbb{G}$.
In Emura et al.' scheme (Emura et al., 2009), the decrypting entity needs to perform two pairing operations and 3 exponentiations in $\mathbb{G}_1$. Herranz et al. (Herranz et al., 2010) decryption algorithm requires $n+1$ exponentiations in $\mathbb{G}_1$, 5 pairing functions and one exponentiation in $\mathbb{G}$. In (Ge et al., 2012) scheme, the users executes 4 pairing operations and $2(k-n)$ exponentiations in $\mathbb{G}_1$. Wang et al. proposed a KP-ABE scheme (Wang and Luo, 2012) where the decryption algorithm performs $n+1$ exponentiations in $\mathbb{G}$, one exponentiation in $\mathbb{G}_1$ and two pairing operations.

The aforementioned schemes do not ensure access policies update. Jiang et al.'s scheme is the first CP-ABE scheme supporting policy updates. In this

Table 1: Features and Functionality Comparison of Attribute Based Encryption Schemes

Scheme	*Policy Update*	*Access Policy*	*Type*	*Key size*	*Ciphertext size*	*Encryption Cost*	*Update Cost*	*Decryption Cost*
(Emura et al., 2009)	✗	AND-Gates	CP-ABE	2	3	$2E_1+E$	–	$2\tau_p+3E_1$
(Herranz et al., 2010)	✗	Monotone	KP-ABE	k+n	3	$2E_1+E$	–	$(n+1)E_1+5\tau_p+E$
(Ge et al., 2012)	✗	Threshold	CP-ABE	$3k-2+n$	4	$5E_1+E$	–	$4\tau_p+2E_1(k-n)$
(Wang and Luo, 2012)	✗	Monotone	KP-ABE	$k(n+1)$	3	$2E_1+E$	–	$(n+1)E+E_1+2\tau_p$
(Jiang et al., 2017)	✓	Threshold	CP-ABE	$n+1$	$3+k-m$ / 3	$(k-m+2)E_1+E$	tE_1	$nE_1+2\tau_p$
	✓	Threshold	CP-ABE	$n+1$	$r+3$ / 3	$(r+2)E_1+E$	$(2t+2)E_1+\tau_p$	$nE_1+2\tau_p$
PU-ABE	✓	Monotone	KP-ABE	$(n+1)k$	$3+k-m+r/3$	$(k-m+2+r)E_1+E$	$tE_1/(2t+2)E_1+\tau_p$	$2\tau_p+E_1+(n+1)E$

We denote by E_1 the exponentiation cost in $\mathbb{G}_1$, E the exponentiation cost in $\mathbb{G}$ and τ_P the computation cost of a pairing function $\hat{e}$. n is the number of attributes of a user. k is the cardinal of the universe of attributes $\mathbb{U}$. m is the number of attributes used in an access policy for encryption. t is the number of attributes ($t = |\mathcal{U}|$) added or removed from an access policy and r is the maximum number of attributes supported by the revocation list.

scheme, the encryption algorithm related to the attributes addition construction require $k-m+2$ exponentiations in $\mathbb{G}_1$ and only one exponentiation in $\mathbb{G}$. In addition, the attribute revocation encryption algorithms requires $r+2$ exponentiations in $\mathbb{G}_1$ and only one exponentiation in $\mathbb{G}$. This proposal consists in executing an update algorithm by the cloud server to update the used access policy used in the encryption. Therefore, it requires t exponentiations in $\mathbb{G}_1$ to add attributes and $2t+2$ exponentiations in $\mathbb{G}_1$ and one pairing operation to revoke attributes. In (Jiang et al., 2017), the decryption algorithm incurs $2\tau_p+nE_1$ overhead.

PU-ABE scheme requires the execution of $(k-m+2+r)$ exponentiations in $\mathbb{G}_1$ and only one exponentiation in $\mathbb{G}$. The proposed scheme requires the execution of two update functions to add attributes or revoke attributes from the access policy. To add new attributes, it requires t exponentiations in $\mathbb{G}_1$ while revoking t attributes needs to an overhead equal to $(2t+2)E_1+\tau_p$, where t is the number of attributes to be added or removed. Our PU-ABE scheme requires $2\tau_p+E_1+(n+1)E$ as a decryption overhead due to the use of monotone access policies.

Above all, our proposed PU-ABE scheme presents quite similar to the computation costs of the related attribute based encryption schemes while providing more practical features mainly related to expressiveness and policy update.

9 CONCLUSIONS

Attribute based encryption is often used to ensure encrypted access control to outsourced data for multi-user settings. That is, in several applications, users are removed and/or added, thus, it require an efficient update of users' access rights.

In this paper, we propose PU-ABE, a novel key policy attribute based encryption technique, with short size ciphertexts that supports access policy update. Indeed, a cloud server can add and/or remove attributes from the encryption access policy without requiring data re-encryption. The proposed PU-ABE scheme is proved to be secure against chosen ciphertexts attacks in the standard model. Finally, a detailed performances analysis showed that PU-ABE ciphertexts are short-sized and independent from the number of attributes used in the access policy which affords low communication and storage costs, compared to most-closely related schemes.

ACKNOWLEDGEMENTS

This research is supported by STRATUS (Security Technologies Returning Accountability, Trust and User-Centric Services in the Cloud), a project funded by the Ministry of Business, Innovation and Employment (MBIE), New Zealand.

REFERENCES

Attrapadung, N., Herranz, J., Laguillaumie, F., Libert, B., De Panafieu, E., and Ràfols, C. (2012). Attribute-based encryption schemes with constant-size ciphertexts. *Theoretical Computer Science*, 422:15–38.

Atwady, Y. and Hammoudeh, M. (2017). A survey on authentication techniques for the internet of things. In *Proceedings of the International Conference on Future Networks and Distributed Systems*, page 8. ACM.

Bacis, E., De Capitani di Vimercati, S., Foresti, S., Paraboschi, S., Rosa, M., and Samarati, P. (2016a). Mix&slice: Efficient access revocation in the cloud. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 217–228. ACM.

Bacis, E., di Vimercati, S. D. C., Foresti, S., Paraboschi, S., Rosa, M., and Samarati, P. (2016b). Access control management for secure cloud storage. In *International Conference on Security and Privacy in Communication Systems*, pages 353–372. Springer.

Belguith, S., Jemai, A., and Attia, R. (2015). Enhancing data security in cloud computing using a lightweight cryptographic algorithm. In *ICAS 2015 : The*

Eleventh International Conference on Autonomic and Autonomous Systems, pages 98–103. IARIA.

Belguith, S., Kaaniche, N., Jemai, A., Laurent, M., and Attia, R. (2016). Pabac: a privacy preserving attribute based framework for fine grained access control in clouds. In *13th IEEE International Conference on Security and Cryptography(Secrypt)*.

Belguith, S., Kaaniche, N., Laurent, M., Jemai, A., and Attia, R. (2017). Constant-size threshold attribute based signcryption for cloud applications. In *SECRYPT 2017: 14th International Conference on Security and Cryptography*, volume 6, pages 212–225.

Belguith, S., Kaaniche, N., Laurent, M., Jemai, A., and Attia, R. (2018). Phoabe: Securely outsourcing multi-authority attribute based encryption with policy hidden for cloud assisted iot. *Computer Networks*, 133:141–156.

Bethencourt, J., Sahai, A., and Waters, B. (2007). Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*.

Boneh, D., Boyen, X., and Goh, E.-J. (2005). Hierarchical identity based encryption with constant size ciphertext. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 440–456. Springer.

Coates, A., Hammoudeh, M., and Holmes, K. G. (2017). Internet of things for buildings monitoring: Experiences and challenges. In *Proceedings of the International Conference on Future Networks and Distributed Systems*, page 38. ACM.

Emura, K., Miyaji, A., Nomura, A., Omote, K., and Soshi, M. (2009). A ciphertext-policy attribute-based encryption scheme with constant ciphertext length. In *International Conference on Information Security Practice and Experience*, pages 13–23. Springer.

Esposito, C. and Ciampi, M. (2015). On security in publish/subscribe services: a survey. *IEEE Communications Surveys & Tutorials*, 17(2):966–997.

Farhan, M., Jabbar, S., Aslam, M., Hammoudeh, M., Ahmad, M., Khalid, S., Khan, M., and Han, K. (2018). Iot-based students interaction framework using attention-scoring assessment in elearning. *Future Generation Computer Systems*, 79:909–919.

Ge, A., Zhang, R., Chen, C., Ma, C., and Zhang, Z. (2012). Threshold ciphertext policy attribute-based encryption with constant size ciphertexts. In *Australasian Conference on Information Security and Privacy*, pages 336–349. Springer.

Ge, C., Susilo, W., Fang, L., Wang, J., and Shi, Y. (2018). A cca-secure key-policy attribute-based proxy re-encryption in the adaptive corruption model for dropbox data sharing system. *Designs, Codes and Cryptography*, pages 1–17.

Goyal, V., Pandey, O., Sahai, A., and Waters, B. (2006). Attribute-based encryption for fine-grained access control of encrypted data. In *The 13th ACM conference on Computer and communications security*.

Herranz, J., Laguillaumie, F., and Ràfols, C. (2010). Constant size ciphertexts in threshold attribute-based encryption. In *International Workshop on Public Key Cryptography*, pages 19–34. Springer.

Ion, M., Russello, G., and Crispo, B. (2012). Design and implementation of a confidentiality and access control solution for publish/subscribe systems. *Computer networks*, 56(7):2014–2037.

Jiang, Y., Susilo, W., Mu, Y., and Guo, F. (2017). Ciphertext-policy attribute-based encryption supporting access policy update and its extension with preserved attributes. *International Journal of Information Security*, pages 1–16.

Kaaniche, N. and Laurent, M. (2017a). Attribute based encryption for multi-level access control policies. In *SECRYPT 2017: 14th International Conference on Security and Cryptography*, volume 6, pages 67–78. Scitepress.

Kaaniche, N. and Laurent, M. (2017b). Data security and privacy preservation in cloud storage environments based on cryptographic mechanisms. *Computer Communications*, 111:120–141.

Li, J., Sha, F., Zhang, Y., Huang, X., and Shen, J. (2017). Verifiable outsourced decryption of attribute-based encryption with constant ciphertext length. *Security and Communication Networks*, 2017.

Liang, K., Au, M. H., Liu, J. K., Susilo, W., Wong, D. S., Yang, G., Yu, Y., and Yang, A. (2015). A secure and efficient ciphertext-policy attribute-based proxy re-encryption for cloud data sharing. *Future Generation Computer Systems*, 52:95–108.

Nkenyereye, L., Park, Y., and Rhee, K. H. (2016). A secure billing protocol over attribute-based encryption in vehicular cloud computing. *EURASIP Journal on Wireless Communications and Networking*, 2016(1):196.

Ogawa, K., Tamura, S., and Hanaoka, G. (2017). Key management for versatile pay-tv services. In *International Workshop on Security and Trust Management*, pages 3–18. Springer.

Sahai, A. and Waters, B. (2005). Fuzzy identity-based encryption. In *EUROCRYPT 2005*.

Wang, C.-J. and Luo, J.-F. (2012). A key-policy attribute-based encryption scheme with constant size ciphertext. In *Computational Intelligence and Security (CIS), 2012 Eighth International Conference on*, pages 447–451. IEEE.

Yao, X., Chen, Z., and Tian, Y. (2015). A lightweight attribute-based encryption scheme for the internet of things. *Future Generation Computer Systems*, 49:104–112.