PHOABE : securely outsourcing multi-authority attribute based encryption with policy hidden for cloud assisted IoT

Belguith, S, Kaaniche, N, Laurent, M, Jemai, A and Attia, R

http://dx.doi.org/10.1016/j.comnet.2018.01.036

PHOABE: securely outsourcing multi-authority attribute based encryption with policy hidden for cloud assisted IOT

Sana Belguith, Nesrine Kaaniche, Maryline Laurent, Abderrazak Jemai, Rabah Attia

PHOABE: SECURELY OUTSOURCING MULTI-AUTHORITY ATTRIBUTE BASED ENCRYPTION WITH POLICY HIDDEN FOR CLOUD ASSISTED IOT

Sana Belguith[1,2]*, Nesrine Kaaniche*[3]*, Maryline Laurent*[3]*, Abderrazak Jemai*[4]*, Rabah Attia*[1]

[1]SERCom Lab, Ecole Polytechnique de Tunisie
Université de Carthage, Tunisie
[2]Telnet Innovation Labs, Telnet Holding
[3]SAMOVAR, CNRS, Télécom SudParis, CNRS, Université Paris-Saclay,
[4]Laboratory LIP2, University of Sciences of Tunis, Tunisia

ABSTRACT

Attribute based encryption (ABE) is an encrypted access control mechanism that ensures efficient data sharing among dynamic group of users. Nevertheless, this encryption technique presents two main drawbacks, namely high decryption cost and publicly shared access policies, thus leading to possible users' privacy leakage.

In this paper, we introduce PHOABE, a Policy-Hidden Outsourced ABE scheme. Our construction presents several advantages. First, it is a multi-attribute authority ABE scheme. Second, the expensive computations for the ABE decryption process is partially delegated to a Semi Trusted Cloud Server. Third, users' privacy is protected thanks to a hidden access policy. Fourth, PHOABE is proven to be selectively secure, verifiable and policy privacy preserving under the random oracle model. Five, estimation of the processing overhead proves its feasibility in IoT constrained environments.

Index Terms— Attribute based encryption, Hidden policy, Decryption outsourcing, Cloud computing, Privacy, Data security

1. INTRODUCTION

The Internet of Things (IoT) refers to connecting different kinds of devices (things), mainly sensors, RFID tags, PDAs or smartphones to build a network. The deployment of these IoT devices is gaining an expanding interest in the academic research and industrial areas as well as in daily life [1] such as smart grid [2], e-health [3], smart city, etc.

Currently, applications based on IoT can be found everywhere. According to Yao et al. [4], IoT is classified into Unit IoT and Ubiquitous IoT categories according to the number of the involved applications or domains [5]. The unit IoT category is involved in a single application, where only one authority is required. However, in the ubiquitous IoT category, IoT is used in cross domain applications, where local, national and industrial IoTs are interacting, thus requiring multiple authorities across domain applications. Both unit IoT and Ubiquitous IoT are becoming popular, and there is a strong need for both of them to handle data processing and sharing among different IoT devices.

The significant growth of involved IoT devices imposes high requirements for data security and privacy preservation. Hence, security problems have become a hurdle in fulfilling the vision for IoT [6, 7].

In IoT applications, data are always transmitted, stored and dynamically shared through the heterogeneous and distributed networks [8]. Consequently, encryption and access control mechanisms are important in order to prevent unauthorized entities from accessing data [9–13].

Attribute based encryption schemes is a promising cryptographic primitive that ensures efficient encrypted access control to outsourced data. Indeed, recently, several attribute based encryption mechanisms have been proposed in literature [14–20].

Most of the proposed attribute based schemes have focused on designing expressive access control policies and providing low communication overheads, through short or constant size ciphertexts [21–23]. Though these solutions present low storage and communication costs, they are still not suitable to be used on resource-constrained devices such as mobile devices and sensors. For instance, the construction of ABE schemes is based on the use of bilinear maps which present expensive computation costs. Moreover, the number of these expensive bilinear operations increases along with the number of attributes involved in the access structure, mainly in the decryption procedure [24]. Hence, the most relevant challenge is to reduce the decryption processing cost of the introduced ABE mechanism while providing fine-grained access control for users [25].

Green et al. [24] proposed, in 2011, the first attribute-based encryption scheme with outsourced decryption. This scheme consists in securely offloading the decryption process of ABE to an external cloud based provider. This solution

ensures that most of the decryption cost can be released from the IoT devices to the cloud.

In most attribute based encryption schemes, the access structure is shared publicly with the related ciphertext. Hence, any user who get the ciphertext can see its content. This exposure of data's access structure will disclose sensitive information about the decryption or encryption party. Meanwhile, in order to avoid disclosing these sensitive information, the access structure should be hidden [26–28].

In addition, in single-authority ABE schemes, a central attribute authority is responsible for managing and issuing all users' attributes and related secret keys. Although this setting facilitates the key management, it can be a bottleneck since central attribute authority is able to achieve a key escrow attack, due to its knowledge of the users' private keys. To solve this problem, many multi-attribute authority ABE schemes have been proposed. These solutions rely on multiple parties to distribute attributes and private keys to users. Such approach offers the scalability for the system even if the number of users becomes important [21, 29, 30].

In this paper, we introduce a novel Policy-Hidden Outsourced Attribute Based Encryption (PHOABE) scheme. Our proposed mechanism is multifold.

First, we extend the original multi-authority CP-ABE scheme proposed by Lewko et al. [29] to support the outsourced decryption in order to better fit processing and communication requirements of resource-constrained devices. For instance, our scheme consists in delegating the expensive computations during the decryption phase, to a Semi Trusted Cloud Server, referred to as STCS.

Second, we apply policy-hidden techniques to ensure users' privacy and access policy confidentiality preservation.

Third, we introduce a secure mechanism consisting in verifying that the partially decrypted ciphertext was correctly generated by the remote cloud server referred to as the verifiability concept.

Fourth, we show that our proposed mechanism is selectively secure, verifiable and policy privacy preserving under the random oracle model.

Paper Organisation – The remainder of this paper is organized as follows. First, Section 2 highlights security considerations and design goals. Then, Section 3 reviews related work and introduces attribute based mechanisms. In Section 4, we describe the system model and the threat model of the system. Afterwards, we detail the framework design and we introduce the construction of our proposed scheme in Section 5. In Section 6, we perform security analysis of PHOABE based on security games. Finally, a theoretical performance analysis is provided in Section 7, before concluding in Section 8.

2. PROBLEM STATEMENT

As e-health systems are witnessing increased popularity, several health organisations are using these systems in order to centralize and share medical data in an efficient way.

Let us consider the following example, where a medical organisation relies on cloud based services to collect and share Electronic Health Records (EHRs) among the medical staff. Note that the medical staff can belong to different organisations such as hospitals, research laboratories, pharmacies, health ministry as well as doctors. The use of a cloud architecture enables the hospital employees to access the data using their smart devices (such as PDAs, smartphones $\cdots$), considered as resource-constrained devices.

Health Insurance Portability and Accountability Act (HIPAA) [31] states that access policies must finely precise different access privileges of authorized users to the shared outsourced data. In fact, a health-care information system based on cloud services is required to protect medical records from unauthorized access. Hence, the system must restrict access of protected data to eligible doctors. For instance, hospital employees, mainly doctors, have to share patients' health information, in order to collaborate with the involved hospital employees to properly prescript treatments. Thus, they usually form dynamic sharing groups with different granted privileges.

As data are always shared through the heterogeneous and distributed networks, the proposed security mechanisms should provide lightweight processing at the client side, while supporting flexible sharing of encrypted outsourced data among dynamic group of users.

To support all these features with efficiency, we propose to design a multi-attribute authority ABE scheme with outsourced decryption to be run at the client side.

Thus, the proposed scheme PHOABE must fulfill the following properties:

- ***low computation overhead –*** PHOABE must introduce cryptographic algorithms with low processing complexity especially at the client side in order to ensure access by different resource-constrained devices.
- ***data confidentiality –*** PHOABE has to protect the secrecy of outsourced and encrypted data contents against both curious cloud service providers and malicious users.
- ***flexible access control –*** our proposal should ensure fine grained access control to allow authorized users to access data.
- ***privacy –*** PHOABE must protect group members' access patterns privacy, while requesting access to outsourced data.

3. ABE-RELATED WORK

Attribute Based Encryption (ABE) was first designed by Sahai and Waters to ensure encrypted access control [32]. In ABE schemes, the ciphertext is encrypted for many users instead of encrypting to a single user as in traditional public key cryptography. In attribute based encryption schemes, user's private keys and ciphertext are associated with an access policy or a set of attributes [33]. Thus, a data user is able to decrypt the ciphertext if his private key matches the ciphertext. ABE schemes are classified into two categories, namely: Key-Policy Attribute Based Encryption (KP-ABE) and Ciphertext-Policy Attribute Based Encryption (CP-ABE) [34].

In KP-ABE, the ciphertext are labeled with a set of attributes while the users' private keys are associated with an access policy which can be any monotonic tree. The user is able to decrypt the ciphertext if its access policy is satisfied by the attributes embedded in the ciphertext. Although the KP-ABE scheme offers fine-grained access control feature, it has one main disadvantage. Indeed, the data owners cannot decide on who has access to their encrypted data, except by their choice of descriptive attributes for the data, as the access policy is embedded in the user's private keys. Consequently, the data owners have to trust the key issuer. Ciphertext-policy ABE schemes remove such inconvenience by directly embedding the access policy on the ciphertext. As such, the data owners can now authorize who can have access on their encrypted data [21, 29, 34].

CP-ABE schemes allow the data owner to precise the users authorized to access the data, by embedding the access policy to the ciphertext [21], [35], [26] [36]. In order to issue private keys related to the user's set of attributes, ABE schemes rely on trusted authorities. ABE schemes can be categorized into two types namely single-authority ABE schemes and multi-authority ABE schemes.

In a single-authority ABE scheme, the attributes and their related private keys are issued by a central attribute authority. Although this centralized approach makes the key management easier, it does not ensure scalability especially when involving a huge number of users. To address this problem, multi-attribute authority ABE schemes [29, 30, 37] have been proposed.

In 2011, Lewko and Waters [29] proposed a multi-attribute authority scheme consisting on issuing attributes and their related secret keys from different attribute authorities. For instance, an attribute authority is responsible for generating a private key associated to a user's attribute. Consequently, this scheme does not rely on a central trusted authority to manage attributes' secret keys. In addition, Lewko and Waters use a Unique Global Identifier (GID) for each user to prevent collision attacks. Hence, a user must send his unique GID to each attribute authority to receive his attribute's secret key. Although CP-ABE ensures fine grained and flexible access control, it requires an expensive decryption costs which be not convenient for resource-constrained devices.

For instance, these expensive computation costs are mainly related to the execution of several pairing functions. In addition, the decryption process requires the execution of a number of pairing operations which increases with the number of attributes involved in the access policy [24]. Additionally, the use of CP-ABE schemes consists in sharing the access structure associated with the ciphertext with the involved authorities which can disclose the attributes in the access policy.

In the following, we present a review of the proposed outsourcing attribute based encryption mechanisms in Section 3.1. Then, we introduce policy hidden attribute based encryption schemes in Section 3.2.

3.1. Outsourcing Attribute Based Encryption

As detailed in the aforementioned section, ABE schemes present expensive decryption costs which increase along with the number of attributes of involved attributes in the access policy. In fact, thanks to their bilinearity properties, ABE schemes usually rely on expensive-computing pairing-based operations. Obviously, this limit is mainly prominent for resource-constrained devices.
To reduce expensive costs, several research works rely on the use of constant attribute based encryption schemes generating ciphertext size and relying on a constant number of bilinear operations [23, 36, 38–41]. However, these schemes consist in using threshold or conjunctive access policies which do not provide the desired expressiveness.

To mitigate this drawback, in 2011, Green et al. [24] proposed a new approach consisting in outsourcing the expensive operations during the decryption phase to a third party. This approach consists in generating a *transformation key* derived from the user's secret key. Then, the user shares this generated transformation key with a semi-trusted cloud server (STCS).
The ciphertext is then submitted to the STCS which uses the transformation key to generate a partially decrypted ciphertext of the same message and sends it to the user. Afterwards, the user can recover the original message using the short ciphertext and his secret key with only one exponentiation operation. Note that the semi-trusted cloud server cannot gain any information about the encrypted message while partially decrypting the ciphertext. In addition, this process helps the user to save the local computation costs.

This new concept proposed by Green et al. [24] is similar to the concept of proxy re-encryption [42–44] where an untrusted proxy is given a re-encryption key that allows it to transform an encryption under Alice's key of m into an encryption under Bob's key of the same m, without allowing the proxy to learn anything about m.

While using attribute based encryption schemes with out-

sourced decryption, the user relies on the use of a semi-trusted third party to partially decrypt the ciphertext. Thus, to ensure data security, the user should be able to verify that the received partially decrypted ciphertext is not altered. For instance, a lazy STCS can return a ciphertext which has been previously computed for the user or a malicious STCS can generate a forged transformation of the ciphertext [45].

Some research works introduced verifiable computation techniques which can be used to construct attribute based encryption schemes with verifiable outsourced decryption [46–48]. The solutions proposed in [46, 48] apply a fully homomorphic encryption chemes [49] which require a huge computation costs.

In 2013, Lai et al. [50] proposed a security model to verify the correctness of a ciphertext generated by an outsourced attribute based encryption scheme. Although the authors introduced a verifiable outsourced attribute based encryption scheme, this latter presents an expensive computation cost at the encrypting entity side which is not suitable for resource-constrained devices. For instance, in the proposed scheme, the ciphertext is composed by the encrypted message and an encrypted random message. Thus, to ensure the verification of the correctness of the partially decrypted ciphertext, the encrypting entity adds a redundant encrypted message in the ciphertext.

Li et al. proposed, in 2014, an attribute based encryption scheme with outsourced decryption while ensuring the ciphertext verifiability [51], based on their first construction presented in [52]. Their proposal requires the deployment of two cloud service providers to perform the outsourced key-issuing and decryption algorithms. However, proposed scheme relies on a single-authority ABE scheme authority. Thus, all the attribute involved in the system are issued and managed using a central attribute authority.

Afterwards, Wang et al. [53] introduced, in 2015, a server aided CP-ABE system. In fact, users can rely on a proxy-server to pre-compute a partially encrypted ciphertext permitting to improve the efficiency of the encryption process. Then, the encrypting entity uses the partially encrypted ciphertext to generate the encryption of the message and share it. In the decryption phase, users rely on a computing server to perform expensive operations introduced by the decryption process. Although this proposal presents a significant reduction of computation costs, it relies on the use of a single domain architecture which is inconvenient for distributed IoT systems.

Qin et al. [45] extend an attribute based encryption scheme with the verifiable outsourcing feature. Although this proposal presents low computation cost, it relies single-authority ABE scheme. Hence, this can be a bottleneck as this authority may achieve a key escrow attack, due to its knowledge of all users' private keys.

In 2015, Lin et al. [54] propose a verifiable oustourced attribute based encryotion scheme. In this proposal, the verification process relies on the use of an attribute based key encapsulation mechanism, a symmetric-key encryption scheme and a commitment scheme. Their construction is based on the ABE scheme proposed by Waters in 2011 [35] which is a single attribute authority ABE scheme.

Zuo et al. [25] present a CCA-secure ABE scheme with outsourced decryption for fog computing applications. This scheme does not introduce a mechanism to verify the correctness of the partially decrypted ciphertext, generated from the outsourcing decryption algorithm.

Recently, in 2017, Li et al. [55] proposed an attribute based encryption scheme with the verifiable outsourced decryption feature. This single-authority construction provides constant-size ciphertexts while relying on the use of monotone access structures.

Above all, the aforementioned schemes rely on single-authority attribute based encryption schemes. However, IoT is used in cross domain applications, where local, national and industrial IoTs are interacting. Consequently, multi-attribute authority ABE schemes are more suitable in IoT context. This motivates us to introduce a verifiable outsourced attribute based encryption scheme with multi-attribute authority construction.

3.2. Policy Hidden Attribute Based Encryption

To support flexible access data control, CP-ABE schemes have been widely applied in distributed architectures. However, access policies are usually publicly shared with the different involved entities, which may disclose sensitive information about both decrypting and encrypting entities.

To protect sensitive information included in access policies, several research works [36, 56] introduces CP-ABE schemes with a partially hidden policies. In fact, an access policy involves a set of attributes expressed as a couple: the generic attribute name and the attribute value. Usually, the attribute value contains more sensitive information. For instance, the attribute values "pediatrician" and "XF12599" are more sensitive than the attribute names "Doctor" and "Patient", respectively. Therefore, CP-ABE schemes with partially hidden policies consists in hiding the attribute value to protect the sensitive information. That is, instead of a full access structure, a partially hidden access structure (e.g., "(Patient: * AND Hospital: *) OR (Doctor: * AND Hospital: *)") which consists of only attribute names without attribute values is attached to a ciphertext. Although ABE schemes with partially hidden access structures ensure attributes' values secrecy, they still suffer from a set of security issues, mainly the off-line dictionary attacks.

In 2008, Nishide et al. [26] proposed an attribute based encryption scheme with partially hidden access control policy. This construction relies on the single authority ABE scheme proposed by Cheung et al. [36]. As such, the [26] proposal uses a central authority to issue attributes and secret keys to

different users.

In 2012, Lai et al. [56] proposed an attribute based encryption scheme with partially hidden access policy. This proposal is based on the Waters' attribute based encryption scheme [35]. Hence, it relies on the use of a single authority architecture which is not convenient for distributed IoT architectures.

To address the security and privacy issues raised by CP-ABE with partially hidden policy schemes, CP-ABE with hidden access policy schemes are introduced [27, 28, 57]. Although these schemes ensure the privacy of access policies, they still suffer from high processing overhead. However, the trade-off between efficiency and perfect privacy is the main design challenge of several security mechanisms.

Xu et al. [28] extended the attribute based encryption scheme proposed by Bethencourt et al. [33] with the hidden access policy feature, for cloud applications. However, this ABE scheme relies on the use of a central attribute authority to manage all the attributes and private keys in the system. Hence, this can be a bottleneck as a central attribute authority is able to achieve a key escrow attack.

In 2015, Zhou et al. [57] proposed a privacy preserving attribute based broadcast encryption scheme. This proposal consists in encrypting a message using an expressive hidden access policy. Then, the encrypted message can be broadcasted with or without explicitly specifying the receivers. However, the [57] construction introduces a high computation cost, such that the deciphering entity needs to perform several pairing operations to decrypt the ciphertext.

Phuong et al. [27] proposed an attribute based encryption scheme with hidden access policy. This construction uses an access policy with only AND gates. Thus, this proposal presents less expressiveness compared to other schemes.

Above all, the mentioned proposals rely on single-authority attribute based encryption schemes. Recently, Zhong et al. [58] proposed the first policy hidden attribute based encryption scheme using multi-attribute authority architecture. However, because of the required pairing operations, this scheme introduces an expensive computation cost at the client side to execute the decryption process.

In most of the existing policy-hidden CP-ABE schemes, The decryption computation costs grow proportionally with complexity of the access structures. This motivates us to introduce a policy hidden multi-attribute authority CP-ABE scheme with decryption outsourcing. To evaluate the objectives given in Section 2, we introduce, in Table 1, a comparison of our scheme PHOABE with different CP-ABE constructions, that are most closely-related to our context. On one hand, several research works have introduced ABE with outsourced decryption scheme [24, 25, 45, 50, 51, 54, 55], PHOABE is the first multi-authority attribute based scheme with outsourced decryption ensuring the ciphertext verifiability. On the other hand, ABE with hidden policy has been addressed [26–28, 57, 58] in several research works, but PHOABE is he only scheme suitable for resource-constrained devices.

3.3. Mathematical Background

In this section, we first introduce the access structure in Section 3.3.1. Then, in Section 3.3.2, we present the bilinear maps. Finally, we introduce some security assumptions.

3.3.1. Access Policies

Access policies can be represented by one of the following formats: boolean functions of attributes or a Linear Secret Sharing Scheme (LSSS) matrix [29].

Definition 1. *Access Structure*

Let $\{P_1, \cdots, P_n\}$ be a set of parties. A collection $\mathtt{A} \subseteq 2^{\{P_1,\cdots,P_n\}}$ 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 $\{P_1, \cdots, P_n\}$, such as $\mathtt{A} \subseteq 2^{\{P_1,\cdots,P_n\}} \setminus \emptyset$.
We note that any access structure can be converted into a boolean function. Boolean functions can be represented by an access tree, where the leaves present the attributes while the intermediate and the root nodes are the logical operators AND ($\wedge$) and OR ($\vee$).

Definition 2. *Linear Secret Sharing Schemes (*LSSS*)*

A Linear Secret Sharing Scheme LSSS [29] over a set of parties P is defined as follows:

1. the shares of each party form a vector over $\mathbb{Z}_p$.
2. Let us consider an $(n \times l)$ matrix A called the share-generating matrix for a Linear Secret Sharing Scheme LSSS. The row $i \in [1, \cdots, n]$ of A is is labeled by a function $\rho(i) : \{1, \cdots, n\} \to \mathbb{P}$. Let $s \in \mathbb{Z}_p$ be a secret value to be shared, then we consider a column vector $\vec{v} = [s, r_2, \cdots, r_n]$ where $r_2, \cdots, r_n \in \mathbb{Z}_p$ are random values. Consequently, $A.\vec{v} = \vec{\lambda}$ is the vector of n shares of the secret s according to LSSS.

3.3.2. Bilinear Maps

Let us consider two multiplicative cyclic groups $\mathbb{G}_1$ and $\mathbb{G}_T$ of prime order P and g a generater of $\mathbb{G}_1$. $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$ is a bilinear map if it fulfill the bilinearity, the non-degeneracy and computability properties. These properties are defined as follows:

1. bilinearity: $\forall u, v \in \mathbb{G}_1$ and $a, b \in \mathbb{Z}_p$, we have $\hat{e}(u^a, v^b) = \hat{e}(u, v)^{ab}$.
2. non-degeneracy: $\hat{e}(g, g) \neq 1$.
3. computability: $\hat{e}$ is efficiently computable. $\hat{e}$ is computed by an efficient algorithm for any $g_1, g_2 \in \mathbb{G}_0$.
   We say that $\mathbb{G}_0$ is a bilinear group if the group operation

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

Scheme	*Type*	*Access Policy*	*Hidden Policy*	*Outsourced Decryption*	*Verifiability*	*Multi-authority*	*Security Models*
[26]	CP-ABE	LSSS	Partially hidden	✗	✗	Single	Selective CPA
[56]	CP-ABE	LSSS	Partially hidden	✗	✗	Single	Selective CPA
[24]	CP-ABE	LSSS	✗	✓	✗	Single	RCCA
[50]	CP-ABE	LSSS	✗	✓	✓	Single	Selective CPA
[51]	CP-ABE	LSSS	✗	✓	✓	Single	RCCA
[57]	CP-ABE	LSSS	Fully Hidden	✗	✗	Single	Selective CPA
[28]	CP-ABE	LSSS	Fully Hidden	✗	✗	Single	Selective CPA
[45]	CP-ABE	LSSS	✗	✓	✓	Single	RCCA
[54]	CP-ABE	LSSS	✗	✓	✓	Single	Selective CPA
[58]	CP-ABE	LSSS	Fully Hidden	✗	✗	Multi	Selective CPA
[27]	CP-ABE	AND gates	Fully Hidden	✗	✗	Single	Selective CPA
[25]	CP-ABE	LSSS	✗	✓	✗	Single	Selective CCA
[55]	CP-ABE	AND gates	✗	✓	✓	Single	RCCA
PHOABE	CP-ABE	LSSS	Fully Hidden	✓	✓	Multi	Selective RCPA

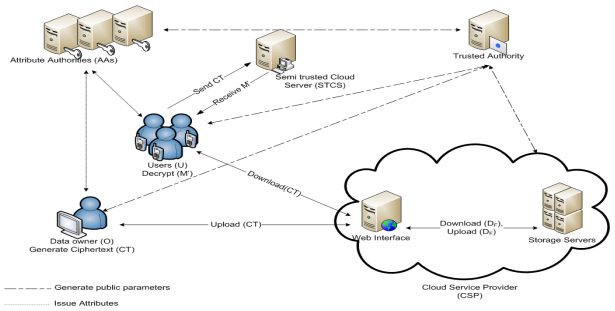

Fig. 1. The PHOABE main architecture entities and their interaction

in $\mathbb{G}_0$ and the bilinear map $\hat{e} : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$ are both efficiently computable.

4. MODEL DESCRIPTION

Our proposed framework considers a cloud storage system involving multiple attribute authorities as detailed in Figure 1. Hence, PHOABE involves five different entities: the Cloud Service Provider (CSP), the Central Trusted Authority (CTA), the Semi-Trusted Cloud Server (STCS), the Attribute Authorities (AA), the data owner (O) and data users (U).

In this section, we introduce our system model in Section 4.1. Then, we detail our security model in Section 4.2.

4.1. System Model

Our PHOABE scheme relies on seven randomized algorithms defined as follows:

$\texttt{setup}(\lambda) \rightarrow \texttt{PP}$ – the setup algorithm is performed by the central trusted authority (CTA) to output the global public parameters PP. Thus, this randomized algorithm takes as input λ which is a chosen security parameter.

$\texttt{setup}_{\texttt{auth}}(\texttt{PP}) \rightarrow (sk_{AA_j}, pk_{AA_j})$ – an attribute authority $AA_j (j \in N)$ executes this randomized algorithm, where N is the number of attribute authorities in the system. The $\texttt{setup}_{\texttt{auth}}$ takes as input PP and generates the pair of private and public keys (sk_{AA_j}, pk_{AA_j}).

$\texttt{encrypt}(\texttt{PP}, \{pk_{AA_j}\}, M, (A, \rho)) \rightarrow CT$ – the encryption algorithm is executed by the data owner (O) to generate the ciphertext CT. It takes as input PP, the set of involved attribute authorities' public keys $\{pk_{AA_j}\}$, the data file M and the access policy $\Psi = (A, \rho)$.

$\texttt{keygen}(\texttt{PP}, sk_{AA_j}, pk_{AA_j}, GID, S_{j,GID}) \rightarrow sk_{j,GID}$ – this algorithm is performed by an attribute authority AA_j in order to generate the user's secret key related to a set of attributes $S_{j,GID} = \{a_{1_j}, \cdots, a_{n_j}\}$, where n_j is the number of attributes of $S_{j,GID}$. It takes as input the global parameters PP, the pair of private and public attribute authority's keys (sk_{AA_j}, pk_{AA_j}) and the users' identity GID. It outputs the secret key $sk_{j,GID}$ related to the set of attributes $S_{j,GID}$.

$\texttt{transform}(\texttt{PP}, \{sk_{j,GID}\}_{j \in N}, (A, \rho), CT) \rightarrow \{tk_{j,GID}\}_{j \in N}$ – this algorithm is performed by the user (U) having a set of attributes S_{GID} and their related secret keys $\{sk_{j,GID}\}_{j \in N}$ received from the different involved attribute authorities $\{AA_j\}_{j \in N}$. It takes as input the global public parameters PP, the user's secret keys $\{sk_{j,GID}\}_{j \in N}$, the access policy $\Psi = (A, \rho)$ and the ciphertext CT. The $\texttt{transform}$ algorithm generates the set of the transformation keys $\{tk_{j,GID}\}_{j \in N} = (\{tpk_{j,GID}\}_{j \in N}, tsk_{GID})$ related to the user's secret keys, where $\{tpk_{j,GID}\}_{j \in N}$ and tsk_{GID} are the public and private transformation keys, respectively.

$\texttt{decrypt}_{\texttt{out}}(\texttt{PP}, \{tpk_{j,GID}\}_{j \in N}, (A, \rho), CT) \rightarrow M'$ – the semi-trusted cloud server (STCS) executes the $\texttt{decrypt}_{\texttt{out}}$ algorithm to retrieve the partially decrypted ciphertext M'.

This algorithm takes as input the public parameters PP, the transformation key $\{tpk_{j,GID}\}_{j\in N}$, an access policy (A,ρ) and the ciphertext CT.

$\texttt{decrypt}(M',tsk_{GID}) \rightarrow M$ – the user U executes the decryption algorithm to retrieve the message M. This algorithm takes as input the transformation secret key tsk_{GID} and the partially decrypted ciphertext M' and outputs the message M.

4.2. Security Model

We consider two realistic threat models for proving security and privacy properties of our PHOABE construction. We first consider a *honest but curious* cloud provider. That is, the cloud is honest as it provides proper inputs or outputs, at each step of the protocol, properly performing any calculations expected from it, but it is curious in the sense that it attempts to gain extra information from the protocol. As such, we consider the honest but curious threat model against the access policy privacy requirement, as presented in Section 4.2.3.
Then, we study the case of malicious users and servers, trying to override their rights. That is, they may attempt to deviate from the protocol or to provide invalid inputs. As such, we consider the malicious user security model against the access policy privacy requirement and the confidentiality property, detailed in Section 4.2.1. Also, we consider the malicious STCS security model against the verifiability requirement, as introduced in Section 4.2.2. For instance, a lazy STCS can return a ciphertext which has been previously computed for the user or a malicious STCS can generate a forged transformation of the ciphertext.

4.2.1. Confidentiality

To design the most suitable security model considering the confidentiality requirement, we adopt a relaxation introduced by Canetti et al. [59] called Replayable CPA (RCPA) security. Indeed, under the RCPA security model, the provided ciphertext can be modified without changing the message in a meaningful way.

In our security model, we assume that the adversary is allowed to query for any secret keys that cannot be used for decrypting the challenge ciphertext. In addition, we consider the assumption introduced in the Lewko et al. proposal [29] that states that the adversary can only corrupt authorities statically. Let consider S_{AA} the set of all attribute authorities and S'_{AA} a set of corrupted attribute authorities.

Our PHOABE scheme is RCCA-Secure if there is no probabilistic polynomial time (PPT) adversary that can win the Exp^{conf} security game defined below with non-negligible advantage.

The Exp^{conf} security game is formally defined, between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ as follows:

Initialisation – in this phase, the adversary $\mathcal{A}$ chooses a challenge access structure $\Psi^* = (A^*,\rho^*)$ and sends it to the challenger $\mathcal{C}$.

Setup – during this phase, the challenger $\mathcal{C}$ first runs the `setup` algorithm to generate the public parameters.

Then, the adversary $\mathcal{A}$ selects a set of corrupted attribute authorities $S'_{AA} \subset S_{AA}$ and runs the $\texttt{setup}_{\texttt{auth}}$ algorithm to obtain their public and private keys.

Subsequently, $\mathcal{C}$ queries the honest attribute authorities' public and private keys by running the $\texttt{setup}_{\texttt{auth}}$ algorithm. Afterwards, the challenger $\mathcal{C}$ publishes the public keys of the honest attribute authorities.

Queries phase 1 – in this phase, the challenger first initializes an empty table T and an empty set D. Then, for each session k, the adversary issues the following queries:

- **Private Key query:** the adversary queries the secret keys $\{sk_{j,GID}\}_{S_{AA}}$ related to a set of attributes $\{S_{GID}\}_k$ belonging to a set of non-corrupted attribute authorities $a_i \in S_{AA} \setminus S'_{AA}$. Then, the challenger sets $D = D \cup \{S_{GID}\}_k$ returns the corresponding secret keys to the adversary. Note that the set of attributes $\{S_{GID}\}_k$ does not satisfy the access policy $\Psi^* = (A^*,\rho^*)$ i.e; $\Psi^*(\{S_{GID}\}_k) \neq 1$.
- **Transformation Key query:** the adversary queries the secret keys $\{sk_{j,GID}\}_{S_{AA}}$ related to a set of attributes $\{S_{GID}\}_k$ belonging to a set of non-corrupted attribute authorities $a_i \in S_{AA} \setminus S'_{AA}$. Afterwards, the challenger searches the entry $(S_{GID},\{sk_{j,GID}\}_{S_{AA}},\{tk_{j,GID}\}_{S_{AA}})$ in table T. If such entry exists, it returns the set of the transformation keys $\{tk_{j,GID}\}_{S_{AA}}$. Otherwise, it generates h used to run the `tranform` . Then, the challenger runs $\texttt{keygen}(\texttt{PP},sk_{AA_j},pk_{AA_j},GID,S_{GID})$ and the $\texttt{transform}(\texttt{PP},\{sk_{j,GID}\}_{j\in N},(A,\rho),CT)$ algorithms and stores in the table T the entry $(S_{GID},\{sk_{j,GID}\}_{S_{AA}},\{tk_{j,GID}\}_{S_{AA}})$. Then, it returns to the adversary the set of the transformation keys $\{tk_{j,GID}\}_{S_{AA}}$.

Challenge – during the challenge phase, the adversary chooses two equal length plaintexts M_0 and M_1 and sends them to the challenger. The challenger $\mathcal{C}$ chooses a random bit b such that $b \in \{0,1\}$ and encrypts CT_b under the access structure (A^*,ρ^*). The generated ciphertext CT_b is then returned to the adversary.

Queries phase 2 – in this phase, the adversary $\mathcal{A}$ who has already received M_b, can query a polynomially bounded number of queries as in **Queries Phase 1**, except that the adversary $\mathcal{A}$ can not query secret keys related to a set of attributes which satisfy the access policy $\Psi^* = (A^*,\rho^*)$.

Guess – the adversary tries to guess which message $M_{b'}$ where $b' \in \{0,1\}$ corresponds to the challenge ciphertext CT_b. The advantage of the adversary to win the game is defined as:

$$Adv_{\mathcal{A}}[Exp^{Conf}(1^{\xi})] = |Pr[b = b'] - \frac{1}{2}|$$

Definition 3. *Our PHOABE scheme is RCPA-Secure ABE (i.e; secure against replayable chosen plaintext attacks)*

against static corruption of the attribute authorities if the advantage $Adv_{\mathcal{A}}[Exp^{Conf}(1^{\xi})]$ *is negligible for all PPT adversaries.*

4.2.2. Verifiability

Our PHOABE scheme is said to be verifiable if there is no probabilistic polynomial time (PPT) adversary that can win the Exp^{verif} security game defined below with non-negligible advantage.

The Exp^{verif} security game is formally defined, between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ as follows:

Initialisation – in this phase, the adversary $\mathcal{A}$ chooses a challenge access structure $\Psi^* = (A^*, \rho^*)$ and sends it to the challenger $\mathcal{C}$.

Setup – during this phase, the challenger $\mathcal{C}$ runs the `setup` algorithm to generate the public parameters.

Then, the adversary $\mathcal{A}$ selects a set of corrupted attribute authorities $S'_{AA} \subset S_{AA}$ and runs the $\texttt{setup}_{\texttt{auth}}$ algorithm to obtain their public and private keys.

Subsequently, $\mathcal{C}$ queries the honest attribute authorities' public and private keys by running the $\texttt{setup}_{\texttt{auth}}$ algorithm. Afterwards, the challenger $\mathcal{C}$ publishes the public keys of the honest attribute authorities.

Queries phase 1 – in this phase, the challenger first initializes an empty table T and an empty set D. Then, for each session k, the adversary issues the following queries:

- **Private Key query:** the adversary queries the secret keys $\{sk_{j,GID}\}_{S_{AA}}$ related to a set of attributes $\{S_{GID}\}_k$ belonging to a set of non-corrupted attribute authorities $S_{AA} \setminus S'_{AA}$. Then, the challenger sets $D = D \cup \{S_{GID}\}_k$ returns the corresponding secret keys to the adversary. Note that the set of attributes $\{S_{GID}\}_k$ does not satisfy the access policy $\Psi^* = (A^*, \rho^*)$ i.e; $\Psi^*(\{S_{GID}\}_k) \neq 1$.
- **Transformation Key query:** the adversary queries the secret keys $\{sk_{j,GID}\}_{S_{AA}}$ related to a set of attributes $\{S_{GID}\}_k$ belonging to a set of non-corrupted attribute authorities $a_i \in S_{AA} \setminus S'_{AA}$. Then, the challenger searches the entry $(S_{GID}, \{sk_{j,GID}\}_{S_{AA}}, \{tk_{j,GID}\}_{S_{AA}})$ in table T. If such entry exists, it returns the set of the transformation keys $\{tk_{j,GID}\}_{S_{AA}}$. Otherwise, it generates h used to run the `transform` algorithm. Then, the challenger runs $\texttt{keygen}(\texttt{PP}, sk_{AA_j}, pk_{AA_j}, GID, S_{GID})$ and the $\texttt{transform}(\texttt{PP}, \{sk_{j,GID}\}_{j \in N}, (A, \rho), CT)$ algorithms and stores in the table T the entry $(S_{GID}, \{sk_{j,GID}\}_{S_{AA}}, \{tk_{j,GID}\}_{S_{AA}})$. Then, it returns to the adversary the set of the transformation keys $\{tk_{j,GID}\}_{S_{AA}}$.

Challenge – during the challenge phase, the adversary chooses a challenge message M^* and sends it to the challenger. The challenger C encrypts M^* and generates the verification key V^* under the access structure (A^*, ρ^*). Then, the generated ciphertext CT^* is returned to the adversary.

Queries phase 2 – in this phase, the adversary $\mathcal{A}$ can query a polynomially bounded number of queries as in **Queries Phase 1**, except that the adversary $\mathcal{A}$ can not query secret keys related to a set of attributes which satisfy the access policy $\Psi^* = (A^*, \rho^*)$.

Forge – the adversary generates an attribute set $\{S^*_{GID}\}$ and a partially decrypted ciphertext M'^* by running the algorithm $\texttt{decrypt}_{\texttt{out}}(\texttt{PP}, \{tpk^*_{j,GID}\}_{j \in N}, (A^*, \rho^*), CT^*)$. We suppose that the tuple $(S^*_{GID}, \{sk^*_{j,GID}\}_{S_{AA}}, \{tk^*_{j,GID}\}_{S_{AA}})$ is included in the table T. Otherwise, the challenger generates the tuple as a response for transformation key query.

The adversary $\mathcal{A}$ wins the game if $\texttt{decrypt}(M'^*, \{tk^*_{j,GID}\}_{S_{AA}}) \notin \{M^*, \perp\}$ and the challenger can verify the generated partially decrypted ciphertext using the verification key V^*.

Hence, the adversary's advantage is defined as follows:

$$Adv_{\mathcal{A}}[Exp^{verif}(1^{\xi})] = |Pr[Exp^{verif}(1^{\xi})] = 1|$$

Definition 4. *Our PHOABE scheme is verifiable, if the advantage* $Adv_{\mathcal{A}}[Exp^{Verif}(1^{\xi})]$ *is negligible for all PPT adversaries.*

4.2.3. Access Policy Privacy Preservation

The notion of access policy privacy consists in hiding the access structure used to encrypt a message from both the server and the decrypting entity. Indeed, the server or any decrypting entity should not be able to gain any knowledge of the policy except that the user knows whether his attributes satisfy the access policy.

Our PHOABE scheme ensures the access policy privacy preservation requirement if there is no probabilistic polynomial time (PPT) adversary that can win the Exp^{Priv} security game defined below with non-negligible advantage.

The Exp^{Priv} security game is formally defined, between an adversary $\mathcal{A}$ and a challenger $\mathcal{C}$ as follows:

Setup – in this phase, the challenger $\mathcal{C}$ first runs the `setup` algorithm to generate the public parameters.

Then, the adversary $\mathcal{A}$ selects a set of corrupted attribute authorities $S'_{AA} \subset S_{AA}$ and runs the $\texttt{setup}_{\texttt{auth}}$ algorithm to obtain their public and private keys.

Subsequently, $\mathcal{C}$ queries the honest attribute authorities' public and private keys by running the $\texttt{setup}_{\texttt{auth}}$ algorithm. Afterwards, the challenger $\mathcal{C}$ publishes the public keys of the honest attribute authorities.

Queries phase 1 – in this phase, the challenger first initializes an empty table T and an empty set D. Then, for each session k, the adversary issues the following queries:

- **Private Key query:** the adversary queries the secret keys related to a set of attributes $\{S_{GID}\}_k$ belonging to a set of non-corrupted attribute authorities $S_{AA} \setminus S'_{AA}$. Then, the challenger sets $D = D \cup \{S_{GID}\}_k$ returns the corresponding secret keys to the adversary. Note that the set of attributes $\{S_{GID}\}_k$ does not satisfy the access policy $\Psi^* = (A^*, \rho^*)$ i.e; $\Psi^*(\{S_{GID}\}_k) \neq 1$.

- **Transformation Key query:** the challenger searches the entry $(S_{GID}, \{sk_{j,GID}\}_{S_{AA}}, \{tk_{j,GID}\}_{S_{AA}})$ in table T. If such entry exists, it returns the set of the transformation keys $\{tk_{j,GID}\}_{S_{AA}}$. Otherwise, it generates h used to run the `tranform` . Then, the challenger runs `keygen`$(\mathtt{PP}, sk_{AA_j}, pk_{AA_j}, GID, S_{GID})$ and the `transform`$(\mathtt{PP}, \{sk_{j,GID}\}_{j\in N}, (A,\rho), CT)$ algorithms and stores in the table T the entry $(S_{GID}, \{sk_{j,GID}\}_{S_{AA}}, \{tk_{j,GID}\}_{S_{AA}})$. Then, it returns to the adversary the set of the transformation keys $\{tk_{j,GID}\}_{S_{AA}}$.

Challenge – during the challenge phase, the adversary $\mathcal{A}$ sends two challenge messages M_1^* and M_2^* and two valid access policies Ψ_1 and Ψ_2 to the challenger under the following restriction: Either all the adversaries satisfy none of the policies Ψ_1 and Ψ_2 or they all satisfy both policies throughout the game. For instance, to ensure that the adversary has not any knowledge of the policy except that he knows whether his attributes satisfy the access policy, the adversary should satisfy either both policies or none of them.

Then, the challenger flips a fair coin $b \in \{1,2\}$ and encrypts a message M_b^* under the access policy Ψ_b according to the sender's encryption algorithm by running the encryption algorithm `encrypt`$(\mathtt{PP}, \{pk_{AA_j}\}, M_b^*, \Psi_b)$. The resulting challenge ciphertext CT^* is then returned to the adversary.

Queries phase – in this phase, the adversary $\mathcal{A}$ can query a polynomially bounded number of queries as in **Queries Phase 1**, except that the adversary $\mathcal{A}$ satisfy none of the policies Ψ_1 and Ψ_2 or they all satisfy both policies throughout the game and $\mathcal{A}$ may not ask the challenger $\mathcal{C}$ for decrypting the challenge ciphertext CT^*.

Guess – the adversary tries to guess which message $M_{b'}^*$ where $b' \in \{1,2\}$ corresponds to the challenge ciphertext CT^*. The advantage of the adversary to win the game is defined as:

$$Adv_{\mathcal{A}}[Exp^{Priv}(1^{\xi})] = |Pr[b = b'] - \frac{1}{2}|$$

Definition 5. *Our PHOABE scheme ensures the access policy privacy preservation against Adaptive Chosen plaintext Attack if the advantage* $Adv_{\mathcal{A}}[Exp^{Priv}(1^{\xi})] = |Pr[b = b'] - \frac{1}{2}|$ *is negligible for all PPT adversaries.*

5. SECURELY OUTSOURCING POLICY HIDDEN ATTRIBUTE BASED ENCRYPTION

5.1. Overview

In this paper, we develop an outsourcing policy hidden multi-authority attribute based encryption scheme as a novel security mechanism for encrypted access control to outsourced data in cloud storage environments. Our proposal is based on the use of the multi-authority attribute based encryption scheme proposed by Lewko et al. [29] in 2011 which has been extended to provide security and functional features such as low computation cost, privacy preservation, fine grained access control and data confidentiality. For instance, our scheme introduces a novel outsourcing attribute based encryption which consists in reducing the decryption cost by securely delegating the most expensive computations in the decryption phase of CP-ABE to a semi-trusted party while ensuring privacy preservation of the access policy. Figure 2 illustrates the general overview of the PHOABE algorithms.

The different notations used in this paper are listed in Table 2.

Table 2. The different notations used in this paper

Notation	Description
CSP	Cloud Service Provider
CTA	Central Trusted Authority
$STCS$	Semi Trusted Cloud Server
AA	Attribute Authority
O	Data Owner
U	User
PP	Public Parameters
sk_{AA_j}	Secret key related to AA_j
pk_{AA_j}	Public key related to AA_j
A	LSSS access matrix
Ψ	Access policy
D_F	Data file
GID	User Global Identifier
$S_{j,GID}$	A set of attributes belonging to a user GID received from AA_j
$sk_{j,GID}$	the secret key related to the set of attributes $S_{j,GID}$, received from AA_j
$\{tk_{j,GID}\}_{j\in N}$	Transformation keys
$(\{tpk_{j,GID}\}_{j\in N}$	Transformation public key
tsk_{GID}	Transformation secret key
M	Data file
CT	The encrypted data file
M'	Partially decrypted data file

5.2. Complexity Assumptions

In our outsourcing policy hidden multi-authority attribute based encryption, we rely on the following complexity assumptions:

Definition 6. ***Computational Diffie Hellman problem (CDH)***

Given a generator g of a multiplicative cyclic group $\mathbb{G}$ of order N and given two group elements $g^a \in \mathbb{G}$ and $g^b \in \mathbb{G}$ where $a, b \in \mathbb{Z}_N$ are two secrets, the problem of calculating g^{ab} from g^a and g^b is called the Computational Diffie Hellman problem.

Definition 7. ***Decisional Bilinear Diffie-Hellman Assumption (DBDH)***

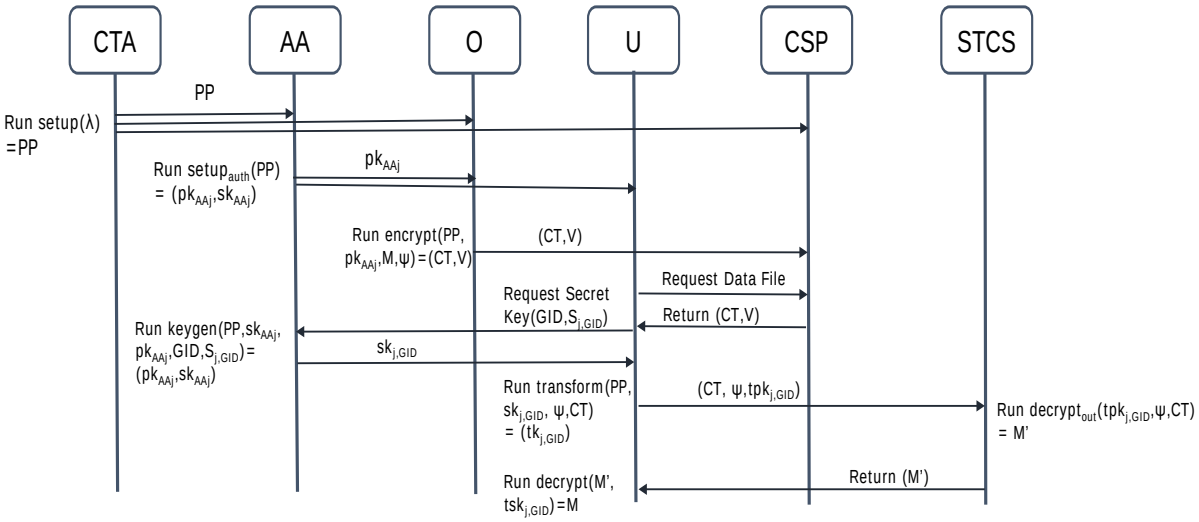

Fig. 2. General Overview of PHAOBE

Given a generator g of a multiplicative cyclic group $\mathbb{G}$ of order N and given three group elements $g^a \in \mathbb{G}$, $g^b \in \mathbb{G}$ and $g^c \in \mathbb{G}$ where $a,b,c \in \mathbb{Z}_N^*$ are three secrets, the problem of distinguishing between tuples of the form $(g^a, g^b, g^c, \hat{e}(g,g)^{abc})$ and $(g^a, g^b, g^c, \hat{e}(g,g)^z)$ for some random integer z, is called the Decisional Bilinear Diffie-Hellman Assumption (DBDH).

5.3. Concrete Construction

Our PHOABE construction is based on seven algorithms defined as follows:

- `setup` – the CTA defines two multiplicative groups $\mathbb{G}_1$ and $\mathbb{G}_T$ of order P, a symmetric bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \to \mathbb{G}_T$, g a generator of $\mathbb{G}_1$ and four collusion resistant hash functions $\mathcal{H}' : \{0,1\}^* \to \mathbb{Z}_P, \mathcal{H}_0 : \{\mathcal{M}\} \to \{0,1\}^{n_{\mathcal{H}_0}}, \mathcal{H}_1 : \{\mathcal{M}\} \to \{0,1\}^*, \mathcal{H}_2 : \{0,1\}^* \to \{0,1\}^{n_{\mathcal{H}_2}}$, where $\mathcal{M}$ is the message universe. Finally, it outputs the global public parameters PP defined as follows:

$$\text{PP} = \{\mathbb{G}_1, \mathbb{G}_T, P, \mathcal{H}', \mathcal{H}_0, \mathcal{H}_1, \mathcal{H}_2, \hat{e}, g\}$$

- $\texttt{setup}_{\texttt{auth}}$ – recall that each attribute authority AA_j, where $j \in \{1, \cdots, N\}$) and N is the number of attribute authorities, manages a set of attributes S_{AA_j}. Each attribute authority chooses two random numbers $\alpha_i, t_i \in \mathbb{Z}_N^*$ for each attribute $i \in S_{AA_j}$ and a number $y_j \in \mathbb{Z}_N^*$. Then, it generates the pair of private and public keys (sk_{AA_j}, pk_{AA_j}) defined as follows:

$$sk_{AA_j} = (\{\alpha_i, t_i\}_{i \in S_{AA_j}}, y_j)$$

$$pk_{AA_j} = (\{\hat{e}(g,g)^{\alpha_i}, g^{t_i}\}_{i \in S_{AA_j}}, g^{y_j})$$

- `encrypt` – this randomized algorithm is based on the following three steps:

(i) First, the data owner O selects a random value $a \in \mathbb{Z}_N^*$ and computes $q_i = \hat{e}((g^{y_j})^a, \mathcal{H}'(x_i))$, where $\{x_i\}_{i \in F}$ denotes one attribute of the access policy Ψ and F is the number of attributes in Ψ. In order to ensure the access policy privacy preservation feature, the data owner replaces each attribute x_i specified in the access structure Ψ, by the computed value q_f. Then, the access policy Ψ is converted to LSSS access matrix $(A_{n \times l}, \rho)$.

(ii) Afterwards, the data owner O picks a random values $s \in \mathbb{Z}_N$. In addition, O selects $p_i \in \mathbb{Z}_N^*$ for each row A_i of A. O chooses a random message $R \in \mathbb{G}_T$. Then, the `encrypt` algorithm computes λ_i and w_i such that $\lambda_i = \vec{A}_i \cdot \vec{v}$, where $\vec{v} = [s, v_1, \cdots, v_l] \in {\mathbb{Z}_N}^l$ is a random vector and $w_i = \vec{A}_i \cdot \vec{\tau}$ such that $\vec{\tau} = [0, \tau_1, \cdots, \tau_l] \in {\mathbb{Z}_N}^l$ is a random vector. The `encrypt` algorithm outputs the ciphertext as a tuple $CT_{ABE} = (h, C_0, C_{1,i}, C_{2,i}, C_{3,i})_{i \in [1,n]}$, where i presents a matrix row corresponding to an attribute i, defined as follows:

$$\begin{cases} h = g^a \\ C_0 = R \cdot \hat{e}(g,g)^s \\ C_{1,i} = g^{\lambda_{\rho(i)}} g^{\alpha_{\rho(i)} p_i} \\ C_{2,i} = g^{p_i} \\ C_{3,i} = g^{t_{\rho_i} p_i} g^{w_i} \end{cases}$$

(iii) Finally, the algorithm sets $R_0 = \mathcal{H}_0(R)$ and computes a symmetric key $K_{sym} = \mathcal{H}_1(R)$. Subsequently, it computes the encryption of the message M using a

symmetric encryption algorithm $\texttt{Encrypt}_{\texttt{sym}}$ such as $CT_{sym} = \texttt{encrypt}_{\texttt{sym}}(K_{sym}, M)$ and the verification key $V = \mathcal{H}_2(R_0||CT_{sym})$.

The `encrypt` algorithm outputs the ciphertext $CT = \{CT_{ABE}, CT_{sym}\}$ as well as the verification key V.

- `keygen` – for any user U having a set of attributes $S_{j,GID} = \{a_{1_j}, \cdots, a_{n_j}\}$ related to an attribute authority AA_j, this latter computes the secret key $sk_{j,GID}$ as follows:

$$sk_{j,GID} = (\{K_{1,i}, K_{2,i}\}_{i \in S_{j,GID}}) = \{g^{\alpha_i} \mathcal{H}'(GID)^{t_i}, \mathcal{H}'(i)^{y_j}\}_{i \in S_{j,GID}}$$

- `tranform` – this randomized algorithm, executed by the decrypting entity, relies on the two following steps:
(i) First, in order to reconstruct the access policy, the user computes the following equation:

$$q'_i = \hat{e}(h, \mathcal{H}'(i)^{y_i}) = \hat{e}(g^a, \mathcal{H}'(i)^{y_j}) \forall i \in S_{j,GID}$$

Then, using q'_i to replace the attribute i, an attribute set S'_{GID} is constructed. The user can identifies the set of attributes $L' = \{i : (\rho(i) \cap S'_{GID})_{i \in [n]}\}$ required for the decryption.
(ii) Second, the user chooses a random value $z \in \mathbb{Z}_N^*$ and sets the transformation keys $\{tk_{j,GID}\}_{j \in N} = (\{tpk_{j,GID}\}_{j \in N}, tsk_{GID})$, where $\{tpk_{j,GID}\}_{j \in N}$ and tsk_{GID} are computed as follows:

$$\begin{cases} \{tpk_{j,GID}\}_{j \in N} = (\{K_{1,i}^{1/z}\}_{i \in L'}, g^{1/z}, H(GID)^{1/z}) \\ \\ tsk_{GID} = z \end{cases}$$

Finally, the user outsources the ciphertext CT and the set of the transformation public keys $\{tpk_{j,GID}\}_{j \in N}$ to the STCS.

- $\texttt{decrypt}_{\texttt{out}}$ – to partially decrypt the ciphertext CT, the STCS proceeds as follows. First, for each matrix row corresponding to an attribute i, the STCS computes:

$$\begin{aligned} ® &= \frac{\hat{e}(g^{1/z}, C_{1,i}).\hat{e}(\mathcal{H}'(GID)^{1/z}, C_{3,i})}{\hat{e}(g^{\alpha_i/z} \mathcal{H}'(GID)^{t_i/z}, C_{2,i})} \\ &= (\hat{e}(g,g)^{\lambda_i} \hat{e}(\mathcal{H}'(GID), g)^{w_i})^{1/z} \end{aligned}$$

Afterwards, the STCS chooses a set of constants $\{c_i\}_{i \in [1,n]} \in \mathbb{Z}_N$ such that $\sum_i c_i \vec{A}_i = [1, 0, \cdots, 0]$. Then, it computes the $\prod_{i=1}^n ®^{c_i}$, such as:

$$\begin{aligned} \prod_{i=1}^n ®^{c_i} &= (\prod_{i=1}^n (\hat{e}(g,g)^{\lambda_i} \hat{e}(g,g)^{w_i t_j})^{1/z} \\ &= (\hat{e}(g,g)^{\sum_{i=1}^n \lambda_i c_i} \hat{e}(g,g)^{\sum_{j=1}^{\kappa} t_j \sum_{i=1}^n w_i c_i})^{\frac{c_i}{z}} \end{aligned}$$

We note that $\lambda_i = \vec{A}.\vec{v}$ and $w_i = \vec{A}.\vec{\tau}$, where $\vec{v}.[1, 0, \cdots, 0] = \sum_{i=1}^n \lambda_i c_i = s$ and $\vec{w}.[1, 0, \cdots, 0] = \sum_{i=1}^n w_i c_i = 0$.

In the sequel, the STCS gets the following result:

$$M' = \prod_{i=1}^n ®^{c_i} = \hat{e}(g,g)^{\frac{s}{z}} \quad (1)$$

Finally, the STCS returns M' to the user.

- `decrypt`– the `decrypt` algorithm includes the following two steps:
(i) First, based on the partially decrypted ciphertext M', the user executes Equation 2 while performing only one exponentiation without calculating any pairing functions to recover the message.

$$R = \frac{C_0}{(R')^{tsk}} = \frac{C_0}{(\hat{e}(g,g)^{\frac{s}{z}})^z} = \frac{C_0}{\hat{e}(g,g)^s} \quad (2)$$

(ii) Then, the user computes $R_0 = \mathcal{H}_0(R)$. If, the algorithm checks $\mathcal{H}_2(R_0||CT_{sym}) \neq V$, then it returns $\perp$ and halts immediately.
Otherwise, it computes $K_{sym} = \mathcal{H}_1(R)$. Then, it returns the message $M = Decrypt_{sym}(K_{sym}, CT_{sym})$.

The proof of correctness of the decryption algorithm is detailed in Section 6.1.

6. SECURITY ANALYSIS

In this section, we first prove the correctness of our PHOABE construction, with respect to the data decryption algorithms, in section 6.1. Then, we prove the security of our proposal, with respect to the indistinguishability, verifiabity and privacy preserving properties, in Section 6.2, 6.3 and 6.4 respectively.

6.1. Correctness

The correctness of our PHOABE construction is detailed by the proof of Lemma 1.

Lemma 1. Data Decryption Correctness.

Proof. (i) After receiving the set of the user's transformation keys related to the involved attributes $\{tpk_{j,GID}\}_{j \in N}$, the STCS first computes:

$$\begin{aligned} ® &= \frac{\hat{e}(g^{1/z}, g^{\lambda_{\rho(i)}} g^{\alpha_{\rho(i)} p_i}).\hat{e}(\mathcal{H}'(GID)^{1/z}, g^{t_{\rho_i} p_i} g^{w_i})}{\hat{e}(g^{\alpha_i/z} \mathcal{H}'(GID)^{t_i/z}, g^{p_i})} \\ &= \frac{\hat{e}(g,g)^{\frac{\lambda_{\rho(i)}}{z}} \hat{e}(g,g)^{\frac{\alpha_{\rho(i)} p_i}{z}} \hat{e}(\mathcal{H}'(GID), g)^{\frac{t_{\rho_i} p_i}{z}} \hat{e}(\mathcal{H}'(GID), g)^{\frac{w_i}{z}}}{\hat{e}(\mathcal{H}'(GID), g)^{\frac{\alpha_{\rho(i)} p_i}{z}} \hat{e}(\mathcal{H}'(GID), g)^{\frac{t_{\rho(i)} p_i}{z}}} \\ &= \hat{e}(g,g)^{\frac{\lambda_{\rho(i)}}{z}} \hat{e}(\mathcal{H}'(GID), g)^{\frac{w_i}{z}} \end{aligned}$$

Then, STCS calculates the constants $c_i \in \mathbb{Z}_N$ such that $\sum_i c_i \cdot \vec{A}_i = [1, 0, \cdots, 0]$.

Note that $\lambda_{\rho(i)} = \vec{A}_{\rho(i)} \cdot \vec{v}$, where $\vec{v} = [s, v_2, \cdots, v_n]$ and $w_i = \vec{A}_i \cdot \vec{\tau}$ such as $\vec{\tau} = [0, \tau_2, \cdots, \tau_n]$. Hence, we deduce that $\sum_{i=1}^{n} \lambda_{\rho(i)} c_i = s$ and $\sum_i w_i c_i = 0$.

Consequently, the partially decrypted ciphertext $M' = \hat{e}(g,g)^{\frac{s}{z}}$ is derived as follows:

$$\begin{aligned}
\prod_{i=1}^{n} \mathbb{R}^{c_i} &= \prod_{i=1}^{n} \hat{e}(g,g)^{c_i \frac{\lambda_{\rho(i)}}{z}} \hat{e}(\mathcal{H}'(GID), g)^{c_i \frac{w_i}{z}} \\
&= \hat{e}(g,g)^{\frac{\sum_{j=1}^{\kappa} c_i \lambda_{\rho(i)}}{z}} \hat{e}(\mathcal{H}'(GID), g)^{c_i \frac{\sum_{j=1}^{\kappa} w_i}{z}} \\
&= \hat{e}(g,g)^{\frac{\sum_{j=1}^{\kappa} c_i \lambda_{\rho(i)}}{z}} \hat{e}(\mathcal{H}'(GID), g)^{\frac{\sum_{j=1}^{\kappa} w_i c_i}{z}} \\
&= \hat{e}(g,g)^{\frac{s}{z}} \hat{e}(\mathcal{H}'(GID), g)^{0} \\
&= \hat{e}(g,g)^{\frac{s}{z}}
\end{aligned}$$

(ii) In the sequel, the user uses the partially decrypted ciphertext M' and the secret transformation key tsk to compute R as follows:

$$R = \frac{C_0}{(R')^{tsk}} = \frac{C_0}{(\hat{e}(g,g)^{\frac{s}{z}})^z} = \frac{C_0}{\hat{e}(g,g)^s} \tag{3}$$

(iii) Then, in order to verify the correctness of the outsourced decryption, the user computes $R_0 = \mathcal{H}_0(R)$.

Then, if he checks $\mathcal{H}_2(R_0 || CT_{sym}) \neq V$,then it returns $\perp$ and halts immediately.

If the decryption is verified, then the user computes the original message as follows:

$$K_{sym} = \mathcal{H}_1(R)$$

.

As such, he retrieves the message as:

$$M = Decrypt_{sym}(K_{sym}, CT_{sym})$$

□

6.2. Indistinguishability

In the following proof, we prove that our scheme is RCPA-Secure against static corruption of the attribute authorities with respect to Theorem 1.

Theorem 1. *If Lewko et al. decentralized CP-ABE scheme [29] is CPA-secure, then, our PHOABE scheme is selectively RCPA-secure such that* $Adv_{\mathcal{A}}[Exp^{conf}] \leq Adv_{\mathcal{A}}[Exp^{Lewko}]$, *according to Definition 3.*

Proof. We define a PPT algorithm adversary $\mathcal{A}$ running the Exp^{conf} security game defined in Section 4.2.1 with an entity $\mathcal{B}$. This entity $\mathcal{B}$ is also running the Lewko et al's CPA-security game (Lewko-Game) with a challenger $\mathcal{C}$. The objective of the proof is to show that the advantage of the adversary $\mathcal{A}$ to win the Exp^{conf} game is smaller than the advantage of the entity $\mathcal{B}$ to win Lewko-Game. Hereafter $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}$ interactions are described, with $\mathcal{A}$ running the following steps and algorithms, as specified in the Exp^{conf} game:

Initialisation – in this phase, the adversary $\mathcal{A}$ gives the algorithm $\mathcal{B}$ a challenge access structure $\Psi^* = (A^*, \rho^*)$.

Setup – $\mathcal{B}$ runs the `setup` to generate the public parameters. For instance, it sets two multiplicative groups $\mathbb{G}_1$ and $\mathbb{G}_T$ of order P, a bilinear map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_T$, g a generator of $\mathbb{G}_1$ and four collusion resistant hash functions $\mathcal{H}'^*, \mathcal{H}^*{}_0, \mathcal{H}^*{}_1, \mathcal{H}^*{}_2 : \{0,1\}^* \rightarrow \mathbb{Z}_P$. Finally, it outputs the public parameters PP defined as $\mathsf{PP} = \{\mathbb{G}_1, \mathbb{G}_T, P, \mathcal{H}'^*, \mathcal{H}^*{}_0, \mathcal{H}^*{}_1, \mathcal{H}^*{}_2, \hat{e}, g\}$. In addition, $\mathcal{B}$ calls the challenger $\mathcal{C}$ to execute the $\mathtt{setup}_{\mathtt{auth}}$ algorithm to generate the attribute authorities public keys. Then, $\mathcal{C}$ chooses two random numbers $\alpha_i, t_i \in \mathbb{Z}_N^*$ for each attribute $i \in S_{AA_j}$ and a number $y_j \in \mathbb{Z}_N^*$. Then, it generates the attribute authorities public keys $\{pk_{AA_j}\}$ defined as $pk_{AA_j} = (\{\hat{e}(g,g)^{\alpha_i}, g^{t_i}\}_{i \in S_{AA_j}}, g^{y_j})$. Finally, $\mathcal{C}$ sends to $\mathcal{B}$ the attribute authorities public keys $\{pk_{AA_j}\}$ which is sent next by $\mathcal{B}$ to $\mathcal{A}$.

Queries phase 1 – $\mathcal{B}$ first initializes an empty table T and an empty set D. Then, for each session k, the adversary issues the following queries:

Private Key query: when the adversary issues a key query by submitting a set of attributes S_{GID} and his identity GID. Then, the algorithm $\mathcal{B}$ uses the challenger $\mathcal{C}$ to generate and return the corresponding secret keys to the adversary $\{K_{1,i}\}_{i \in S_{j,GID}}) = \{g^{\alpha_i} \mathcal{H}'(GID)^{t_i}\}_{i \in S_{j,GID}}$. Afterwards, the challenger $\mathcal{C}$ chooses $y_j \in \mathbb{Z}_N^*$ and sets $\{sk_{j,GID}\}_{j \in N} = (\{K_{1,i}, K_{2,i}\}_{i \in S_{j,GID}}) = \{g^{\alpha_i} \mathcal{H}'(GID)^{t_i}, \mathcal{H}'(i)^{y_j})\}_{i \in S_{j,GID}}$. The secret keys $\{sk_{j,GID}\}_{j \in N})$ are returned to the adversary $\mathcal{A}$. Then, $\mathcal{B}$ sets $D = D \cup \{S_{GID}\}_k$ and returns the secret keys $\{sk_{j,GID}\}_{j \in N}$ to the adversary $\mathcal{A}$.

Transformation Key query: $\mathcal{B}$ searches the entry $(S_{GID}, sk_{S_{GID,j}}, tk_{GID,j})$ in table T. If such entry exists, it returns the transformation key $tk_{GID,j}$ to the adversary $\mathcal{A}$.

Otherwise, it generates a random value a and sets the value $h = g^a$ to simulate the output of the `encrypt` algorithm used to run the `tranform`. h is, then, sent to the challenger $\mathcal{C}$. Then, the challenger chooses a random exponent $z \in \mathbb{Z}_N^*$ and sets the transformation key such as $\{tpk_{j,GID}\}_{j \in N} = (\{K_{1,i}^{1/z}\}_{i \in L'}, g^{1/z}, H(GID)^{1/z})$. Then, $\mathcal{B}$ stores in the table T the entry $(S_{GID}, sk_{GID,j}, tk_{GID,j})$. Then, it returns to the adversary the transformation key $\{tk_{j,GID}\}_{j \in N} = (\{tpk_{j,GID}\}_{j \in N}, tsk_{GID})$.

Challenge – the adversary $\mathcal{A}$ sends two different equal length messages $\{R_0, R_1\} \in \mathbb{G}_T$ to the algorithm $\mathcal{B}$ who forwards them to the challenger $\mathcal{C}$. This latter chooses a bit $b \in \{0,1\}$ and sends R_b to the challenger $\mathcal{C}$. Afterwards, $\mathcal{C}$ chooses a bit $b \in \{0,1\}$ and encrypts R_b under the challenge access structure $\Psi^* = (A^*, \rho^*)$ using Lewko and Waters scheme and returns $CT_{b,ABE}$ to $\mathcal{B}$.

Afterwards, with the probability equal to the advantage of the adversary in Lewko-game ($Adv_{\mathcal{A}}[Exp^{Lewko}]$), $\mathcal{B}$ tries to

guess b. Then, this latter sets $R^*_{0,b} = \mathcal{H}_0(R^*_b)$ and computes a symmetric key $K^*_{sym} = \mathcal{H}_1(R^*_b)$. Consequently, $\mathcal{B}$ encrypts R^*_b using $\texttt{encrypt}_{\texttt{sym}}(K^*_{sym}, R^*_b)$ to generate $CT^*_{b,sym}$. In addition, $\mathcal{B}$ sets the verification key $V^* = \mathcal{H}_2(R^*_{0,b}||CT^*_{b,sym})$. Afterwards, the algorithm $\mathcal{B}$ sends $CT^*_b = (CT^*_{b,ABE}, CT^*_{b,sym})$ to the adversary as a challenge ciphertext as well as the verification key V^*.

Queries phase 2 – $\mathcal{A}$ continues to query a polynomially bounded number of queries and $\mathcal{B}$ answers as in **Queries Phase 1**, except that the adversary $\mathcal{A}$ can not query secret keys related to a set of attributes which satisfy the access policy $\Psi^* = (A^*, \rho^*)$.

Guess – the adversary $\mathcal{A}$ outputs a bit b'. Then, $\mathcal{B}$ sends b' to $\mathcal{C}$ as its guess about b. If $b' = b$, $\mathcal{C}$ answers 1 as the solution to the given instance of the DBDH problem with respect to Definition 7 as introduced in Lewko et al. security analysis [29].

The adversary $\mathcal{A}$ outputs a bit b'. The probability to break the instance of Exp^{conf} game is smaller than the Lewko-Game, as it is necessary for $\mathcal{B}$ to win the game for $\mathcal{A}$ to be able to get the right CT^*_b values, and try to guess the value of b. As such $Pr[Exp_{\mathcal{A}}{}^{Lewko}(1^{\xi})] \geq Pr[Exp_{\mathcal{A}}{}^{conf-real}(1^{\xi})]$, and the advantage of adversary A is negligible. Then our PHOABE scheme satisfies the confidentiality property.

□

6.3. Verifiability

In this section, we prove that our scheme is verifiable against malicious servers with respect to Theorem 2.

Theorem 2. *If $\mathcal{H}_2$ and $\mathcal{H}_0$ are two collision-resistant hash functions, then, our PHOABE scheme is verifiable against malicious servers.*

Proof. We define a PPT algorithm $\mathcal{B}$ which aims to break the verifiability property of the outsourcing ABE system under the help of the adversary $\mathcal{A}$. $\mathcal{B}$ simulates the adversary views with respect to the Exp^{verif} security game defined in Section 4.2.1. To do so, $\mathcal{B}$ tries to break the collision resistance of the $\mathcal{H}_2$ or the $\mathcal{H}_0$ hash functions . Given two challenge hash functions ($\mathcal{H}^*_2$, $\mathcal{H}^*_0$), $\mathcal{B}$ simulates the security game introduced in Definition 4 as follows:

First, $\mathcal{B}$ runs the `setup` algorithms to generate the public parameters except the hash functions $\mathcal{H}_2$ or $\mathcal{H}_0$. In addition, $\mathcal{B}$ generates the attribute authorities public and secret keys by running the $\texttt{setup}_{\texttt{auth}}$ algorithm. Afterwards, $\mathcal{B}$ runs the adversary queries **Queries phase 1** and **Queries phase 2** in order to get secret keys and the transformation keys related to a set of attributes.

In the challenge phase, the adversary $\mathcal{A}$ sends a challenge message M^* to the algorithm $\mathcal{B}$. Then, this latter picks a random message $R^* \in \mathbb{G}_T$ and encrypts R^* under the challenge access structure $\Psi^* = (A^*, \rho^*)$ using Lewko and Waters scheme. Then, $\mathcal{B}$ sets $R^*_0 = \mathcal{H}^*_0(R^*)$ and computes a symmetric key $K^*_{sym} = \mathcal{H}^*_1(R^*)$. Subsequently, it computes the encryption of the message M^* using a symmetric encryption algorithm $\texttt{Encrypt}_{\texttt{sym}}$ such as $CT^*_{sym} = \texttt{encrypt}_{\texttt{sym}}(K^*_{sym}, M^*)$. In addition, $\mathcal{C}$ sets the verification key $V^* = \mathcal{H}^*_2(R^*_0||CT^*_{sym})$. Afterwards, the algorithm $\mathcal{B}$ sends $CT^* = (CT^*_{ABE}, CT^*_{sym}) = (h, C_0, C_{1,i}, C_{2,i}, C_{3,i}, CT^*_{sym})$ to the adversary as a challenge ciphertext as well as the verification key V^*.

If $\mathcal{A}$ breaks the verifiability game, $\mathcal{B}$ will recover a message $M \notin \{M^*, \perp\}$ relying on the partially decryption algorithm $\texttt{decrypt}_{\texttt{out}}(\texttt{PP}, \{tpk_{j,GID}\}_{j \in N}, (A^*, \rho^*), CT^*)$.

Notice that the decryption algorithm outputs $\perp$ if $\mathcal{H}^*_2(R^*_0||CT^*_{sym}) \neq V^*$ where $R^*_0 = \mathcal{H}^*_0(R^*)$ and $R^* = \texttt{decrypt}_{\texttt{sym}}(K^*_{sym}, CT^*_{sym})$. As a consequence, the following two cases are considered:

- **Case 1:** Since $\mathcal{B}$ knows (R^*_0, CT^*_{sym}), if $(R_0, CT_{sym}) \neq (R^*_0, CT^*_{sym})$ is returned as a result, then $\mathcal{B}$ obtains a collision of the hash function $\mathcal{H}^*_2$.
- **Case 2:** If we get $(R_0, CT_{sym}) = (R^*_0, CT^*_{sym})$, while R^* and R are not equal ($R^* \neq R$). Then, $\mathcal{B}$ breaks the collision resistance condition of $\mathcal{H}^*_0$ as $\mathcal{H}^*_0(R) = R_0 = R^*_0 = \mathcal{H}^*_0(R^*)$.

Consequently, using an *absurdum* reasoning, since the hash functions $\mathcal{H}_2$ and $\mathcal{H}_0$ are two collision resistant functions, then our scheme PHOABE is verifiable.

□

6.4. Policy Privacy Preservation

In this section, we prove that our scheme is privacy-preserving against both malicious users and servers with respect to Theorem 3.

Theorem 3. *Our PHOABE scheme is policy privacy preserving according to Definition 5.*

Sketch of proof. In PHOABE scheme, the encrypting entity encrypts data under an access policy Ψ. Afterwards, he generates a new value equal to $\hat{e}((g^{y_j})^a, \mathcal{H}'(x_i))$ based on the one-way anonymous key agreement protocol [60] where a is a random number. This generated value is used to obfuscate the attribute x_i. For instance, in order to ensure the access policy privacy preservation feature, the encrypting entity replaces each attribute x_i specified in the access policy Ψ, by the generated value $\hat{e}((g^{y_j})^a, \mathcal{H}'(x_i))$.

In order to recover the access policy, the decrypting entity uses his corresponding private key $\{K_{2,i}\}_{i \in S_{j,GID}} = \{\mathcal{H}'(i)^{y_j})\}_{i \in S_{j,GID}}$ to compute the value $q'_i = \hat{e}(h, \mathcal{H}'(i)^{y_i}) = \hat{e}(g^a, \mathcal{H}'(i)^{y_i}) \forall i \in S_{GID}$. Hence, only authorized users having the right secret keys can recover the access policy. For instance, thanks to the random value a, unauthorized users cannot guess x_i from the obfuscated value $\hat{e}((g^{y_j})^a, \mathcal{H}'(x_i))$

Consequently, the policy privacy preservation requirement is ensured thanks to the security of the one-way anonymous key agreement protocol [60]. In addition, users cannot

reveal the embedded attribute in the access policy when they collude, because they cannot infer the attribute x_i from $\hat{e}(g^a, \mathcal{H}'(i)^{y_i})$.

7. PERFORMANCE ANALYSIS

In this section, we present the computation and storage complexities of our PHOABE scheme. For this purpose, we are interested by the computations performed at the data owner side in order to execute the `encrypt` algorithm. In addition, we consider the computation cost related to the execution of `decrypt` and $\texttt{decrypt}_{\texttt{out}}$ algorithms performed by the user (U) and the Semi Trusted Cloud Server (STCS), respectively. Moreover, we introduce the size of the user's secret keys as well as the ciphertext's size.
For this purpose, we denote by:

- E_1 : exponentiation in $\mathbb{G}_1$
- E_T : exponentiation in $\mathbb{G}_T$
- τ_P : computation of a pairing function $\hat{e}$
- n is number of attributes used in the access policy
- $|S|$ is the number of attributes in the set of attributes related to a user

Table 3 details the performance comparison with the most closely related ABE schemes.

7.1. Computation Complexities

In [26], Nishide et al. proposed a hidden policy ABE scheme requiring one exponentiation in $\mathcal{G}_T$ and $(2n+1)$ exponentiations in $\mathcal{G}_1$ in the encryption process. For the decryption phase, the user needs to compute $(2n+1)$ pairing functions. In 2011, Green et al. [24] proposed the first ABE scheme with outsourced decryption. This proposal consists in using a semi trusted server to compute $(2n+1)$ pairing functions and $2n$ exponentiation in $\mathcal{G}_T$ in order to generate a partially decrypted ciphertext. Afterwards, the user needs only one exponentiation in $\mathcal{G}_T$ to retrieve the original message. In the encryption process, the data owner needs $(4+3n)$ exponentiations in $\mathcal{G}_1$ and one exponentiation in $\mathcal{G}_T$. Lai et al. [50] introduced an outsourced ABE scheme consisting in verifying the correctness of the ciphertext. In this scheme, the encryption algorithm requires 2 exponentiations in $\mathcal{G}_T$ and the decryption algorithm performs $6n$ exponentiations in $\mathcal{G}_1$. Beyond the encryption and decryption costs, this scheme consists in computing 2 hash functions, in both the encryption and decryption processes, in order to use them in the verification process. Similarly, Qin et al. [45] and Lin et al. [54] proposed two verifiable outsourced ABE schemes. In the [45] proposal, the encryption and decryption algorithms overheads are equal to $\tau_p + E_1(4+3n)$ and $2nE_T + \tau_p(2n+1)$, respectively. To verify the ciphertext's correctness, 3 hash functions have to be performed at both the data owner and user sides. In the [54] construction, the data owner needs to compute $(3+2n)$ exponentiations in $\mathcal{G}_1$, one exponentiation in $\mathcal{G}_T$ and 2 hash functions. Moreover, the user computes one exponentiation in $\mathcal{G}_T$ and 2 hash functions to decrypt and verify the ciphertext. In addition, the semi trusted server computes $2n$ exponentiations in $\mathcal{G}_T$ and $(2n+1)$ pairing functions. In the outsourced attribute based encryption scheme proposed by Li et al. [51], the user needs to compute only one exponentiation in $\mathcal{G}_T$ while outsourcing $(2n+2)$ pairing functions and $(2n+2)$ exponentiations in $\mathcal{G}_T$. The encryption algorithm performs $(3+2n)$ exponentiations in $\mathcal{G}_1$. In 2015, Zhou et al. [57] introduced a policy hiding broadcast ABE scheme where the encryption and decryption overheads are independent from the number of attributes involved in the access structure. Afterwards, Xu et al. [28] proposed a policy privacy preserving ABE scheme. The processing costs are equal to $E_1(n+2)$ and $\tau_p(n+1)+nE_T$ for the encryption and decryption phases, respectively. In 2016, Zhong et al. [58] proposed the first multi-attribute authority ABE scheme with hidden access policy. This proposal consists in performing 2 pairing functions, $3n$ exponentiations in $\mathcal{G}_1$ and $(1+2n)$ exponentiations in $\mathcal{G}_T$ in the encryption phase. Moreover, the decryption algorithm computes $(1+2n)$ pairing functions and n exponentiations in $\mathcal{G}_T$. Zho et al. [25] proposed an outsourced ABE scheme for fog computing applications. This scheme consists in computing $(N+1)$ exponentiations in $\mathcal{G}_1$, 4 exponentiations in $\mathcal{G}_T$, where N is the cardinal of the attribute universe, and 2 hash functions to decrypt the message. The semi trusted server partially decrypts the message by performing $(4+2N+n)$ pairing functions, N exponentiations in $\mathcal{G}_T$ and 2 hash functions. Finally, the user has to perform only 4 exponentiations in $\mathcal{G}_T$ and 2 hash functions to decrypt data. Recently, Li et al. [55] proposed a verifiable outsourced ABE scheme with constant ciphertext size. In this proposal, to encrypt data, the data owner performs 6 exponentiations in $\mathcal{G}_1$, 2 exponentiations in $\mathcal{G}_T$ and one hash function. In addition, the semi trusted server computes 4 pairing functions and the user performs one exponentiation in $\mathcal{G}_T$ and 2 hash functions.

In our PHOABE scheme, the data owner performs 5 exponentiations in $\mathcal{G}_1$, one exponentiation in $\mathcal{G}_T$ and 3 hash functions. In addition, our proposal consists in using a semi trusted cloud server to compute the expensive computation operations such as performing $3n$ pairing functions and one exponentiation in $\mathcal{G}_T$. As a consequence, the user only computes one exponentiation in $\mathcal{G}_T$ and 3 hash functions to execute the verification process.

Above all, the processing costs of our PHOABE scheme remain competitive compared to other encryption schemes. Indeed, our construction consists in outsourcing the computation of expensive operations.

Table 3. Computation and Storage Costs' Comparison of Attribute Based Encryption Schemes

Scheme	*outourced decryption*	*Key Size*	*Transform Key Size*	*Ciphertext Size*	*Encryption Cost*	*User Decryption Cost*	*STCS Decryption Cost*
[26]	✗	$1+\|S\|$	–	$2+2n$	$E_T+E_1(2n+1)$	$\tau_p(2n+1)$	–
[24]	✓	$2+\|S\|$	$3+\|S\|$	$4+2n$	$E_T+E_1(4+3n)$	E_T	$2nE_T+\tau_p(2n+1)$
[50]	✓	$2+\|S\|$	$3+\|S\|$	$5+4n$	$E_T+2E_1+2O(\mathcal{H})$	$2E_T+4E_1+6nE_1+2O(\mathcal{H})$	$\tau_p(2+4n)+4nE_T$
[51]	✓	$2\|S\|$	$2\|S\|+1$	$4+n$	$E_1(3+2n)$	$\tau_p(2n+2)+E_T(2+2n)$	E_T
[57]	✗	$1+\|2S\|$	–	4	$3E_1$	$4\tau_p$	–
[28]	✗	$1+\|S\|$	–	$2+n$	$E_1(n+2)$	$\tau_p(n+1)+nE_T$	–
[45]	✓	$3+\|S\|$	$4+\|S\|$	$4+2n$	$E_T+\tau_p+E_1(4+3n)3O(\mathcal{H})$	$E_T+3O(\mathcal{H})$	$2nE_T+\tau_p(2n+1)$
[54]	✓	$2+\|S\|$	$3+\|S\|$	$4+n$	$E_1(1+4n)+E_T+2O(\mathcal{H})$	$E_T+2O(\mathcal{H})$	$2nE_T+\tau_p(2n+1)$
[58]	✗	$2\|S\|$	–	$3+3n$	$2\tau_p+E_T(1+2n)+3nE_1$	$\tau_p(1+2n)+nE_T$	–
[27]	✗	6	–	5	$8E_1$	$6\tau_p+4nE_1$	–
[25]	✓	$N+\|S\|+1$	$N+2$	$5+N$	$E_T+E_1(N+1)+2O(\mathcal{H})$	$4E_T+4O(\mathcal{H})$	$\tau_p(4+2N+n)+NE_T+O(\mathcal{H})$
[55]	✓	2	3	8	$6E_1+2E_T+O(\mathcal{H})$	$4\tau_p$	$E_T+2O(\mathcal{H})$
PHOABE	✓	$2\|S\|$	$2\|S\|+3$	$4+3n$	$5E_1+E_T+3O(\mathcal{H})$	$E_T+3O(\mathcal{H})$	$3n\tau_p+E_T$

7.2. Storage Complexities

The policy hidden attribute based encryption schemes [26,28] have the same size of user's secret keys which is equal to $1+|S|$, where S represents the set of attributes of the user. Differently, the proposals introduced in [57, 58] have almost the same size of user's secret keys which is equal to $|2S|$. The attribute based encryption schemes with outsourced decryption, presented in [24, 45, 50, 54] consist in using user's secrets keys and transformation keys. Consequently, the total size of user's keys is equal to double the size of his set of attributes. The ABE schemes [27, 55] have constant size of user's secret keys. However, since Li et al. [55] proposal is an ABE scheme with outsourced decryption, the user's secret keys involve the transformation keys whose size is equal to 8. The size of the user's keys in the Zuo's et al. proposal [25] are equal to $N+|S|+1$ and $N+2$ for the user's secret keys and the transformation keys, respectively. Note that N is the cardinal of the attribute universe.

In both Li et al. [51] and PHOABE proposals, the size of the user's secret keys is equal to $2|S|$ while the size of the transformation keys is equal to $2|S|+3$.

The ABE schemes [27, 55, 57] present a ciphertext with a constant size which does not depend on the number of attributes used in the access policy. In the existing ABE schemes such as [24, 26, 45], the authors introduce ABE schemes where the size of the generated ciphertext is approximately equal to $2n$. Recall that n is the number of attributes involved in the access policy. In [28], Xu et al. introduce a hidden policy ABE scheme with a ciphertext's size equal to $2+n$. In 2016, Zuo et al. proposed an outsourced ABE scheme for fog computing applications where the ciphertext's size depends on the cardinal of the set of attributes. The ciphertext size of the ABE schemes with outsourced decryption proposed by Li et al. and Lin et al. in [51, 54] is proportional to $4+n$. The ABE technique with outsourced decryption proposed by Lai et al. presents a ciphertext's size equal to $5+4n$. The multi-authority ABE scheme proposed by Zhong et al. in [58] introduces a ciphertext's size equivalent to 3 times the cardinal of the access policy. Similarly, our PHOABE construction produces a ciphertext which is 3 times the cardinal of the access structure.

Taking into consideration that PHOABE is a multi-authority scheme, we state that our construction presents interesting performances especially related to the key size and the size of the ciphertext.

7.3. Resource-Constrained Performance Analysis

Several research works have been proposed to evaluate the computation overhead of attribute based encryption schemes on resource-constrained devices [61–65]. These papers introduced experimental performance analysis of the different ABE algorithms on mobile devices and sensors.

Ometov et al. [64] evaluate the impact of elementary cryptographic operations on different resource-constrained devices as detailed in Table 4. As our PHOABE framework relies on the use of bilinear maps as well as mathematical operations in a multiplicative group, we investigate the impacts of these operations (Figures 3, 4, and 5) on the performance of different IoT devices, based on the results introduced in [64].

Figure 5 shows the computation cost of one pairing operation in different IoT devices (cf., Table 4). The most efficient device here is Intel Edison with JDK 1.8.0 that computes a single pairing operation in around 500 ms.

As the pairing operations are extremely expensive, attribute based encryption can not be applied to secure IoT devices. Thus, the only solution to benefit from the security feature of ABE is to apply it using outsourced decryption feature.

In ciphertext attribute based encryption, the size of an encrypted file mainly depends on the number of attributes involved in the access policy used in the encryption phase [39].

To estimate the real-world computation costs on a client side, we investigate the computation overhead of PHOABE on a Samsung I9500 Galaxy S4. As shown in Figure 6, the computation costs at the user side increase with the number of attributes used in the encryption.

However, while applying outsourced decryption, the user computation costs are independent of the number of attributes as the user only needs to compute one exponentiation and one

Table 4. Selected Devices [64]

Device	Type	Processor
Sony SmartWatch 3 SWR50	Smart Watch	520 MHz Single-core Cortex-A7
Samsung I9500 Galaxy S4	Smartphone	1.6 GHz Dual-Core Cortex-A15
Jiayu S3 Advanced	Smartphone	1.7 GHz Octa-Core 64bit Cortex A53
Intel Edison	IoT Development Board	500 MHz Dual-Core Intel AtomTM CPU, 100 Mhz MCU
Raspberry Pi 1 model B	IoT Development Board	700 MHz Single-Core ARM Cortex-A6
Raspberry Pi 2 model B	IoT Development Board	900 MHz Quad-Core ARM Cortex-A7

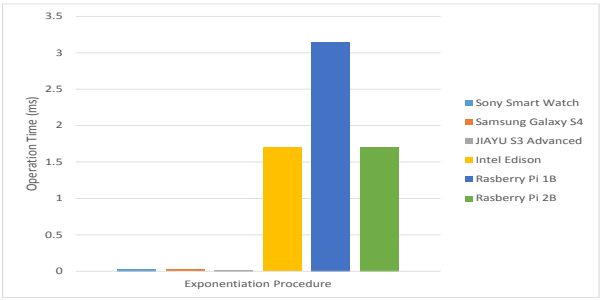

Fig. 3. Exponentiation Execution Time [64]

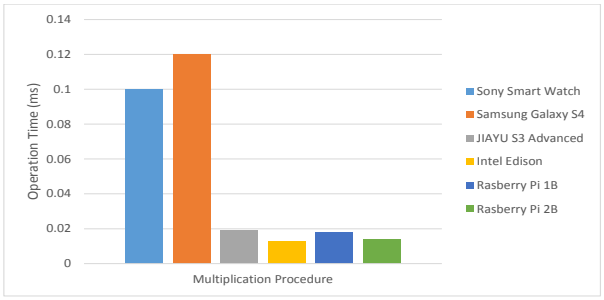

Fig. 4. Multiplication Execution Time [64]

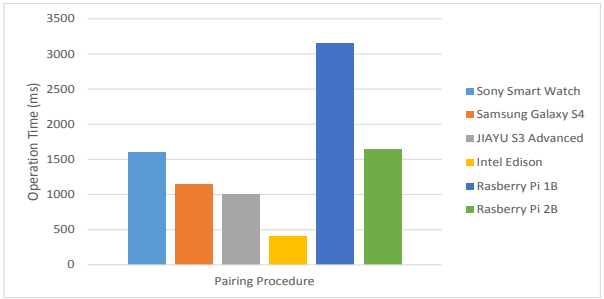

Fig. 5. Pairing Execution Time Curve Type A [64]

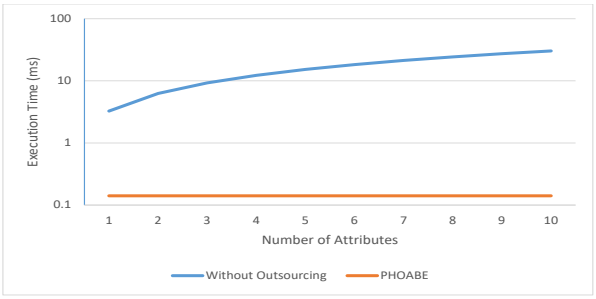

Fig. 6. Computation Costs at the User Side Using PHOABE

multiplication (c.f, Table 5).

Table 5. Number of Cryptographic Operations computed by STCS and User in PHOABE.

	Exp	Mul	Pairing
STCS	1	2	3n
USer	1	1	0

8. CONCLUSION

The number of devices connecting to the Internet of Things (IoT) is growing exponentially, as does the amount of data produced. As such, cloud assisted IoT services is emerging as a promising solution to deal with the computation and storage of the huge amounts of produced data and to delegate the expensive operations to cloud servers. The widespread use of these services requires customized security and privacy levels to be guaranteed.

In this paper, we present PHOABE, a novel privacy-preserving outsourcing multi-authority attribute based encryption scheme, that permits to overcomes the computational costs of decryption that scale with the complexity of the access policy and the number of attributes. The proposed technique can be considered as an optimization of the decryption algorithm to mitigate the practical issues in implementing CP-ABE on resource-constrained devices. For instance, PHOABE is a multi-attribute authority mechanism

which ensures delegating the expensive computations for ABE decryption to a Semi Trusted Cloud Server and protecting user's privacy using a hidden access policy. Finally, PHOABE is proven to be selectively secure, verifiable and policy privacy preserving under the random oracle model.

As for future work, we plan to further improve PHOABE by exploring direct revocation solution to achieve more security features. For instance, a compromised user should be revoked without affecting other users.

ACKNOWLEDGEMENTS

This work is part of the MOBIDOC project achieved under the PASRI program, funded by the European Union and administered by the ANPR.

REFERENCES

[1] Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, and Marimuthu Palaniswami, "Internet of things (iot): A vision, architectural elements, and future directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645–1660, 2013.

[2] Miao Yun and Bu Yuxin, "Research on the architecture and key technology of internet of things (iot) applied on smart grid," in *Advances in Energy Engineering (ICAEE), 2010 International Conference on*. IEEE, 2010, pp. 69–72.

[3] Abdelrahman Abuarqoub and Mohammad Hammoudeh, "Behaviour profiling in healthcare applications using the internet of things technology," *International Journal of Advances in Computer Science & Its Applications*, vol. 6, no. 3, 2016.

[4] Xuanxia Yao, Zhi Chen, and Ye Tian, "A lightweight attribute-based encryption scheme for the internet of things," *Future Generation Computer Systems*, vol. 49, pp. 104–112, 2015.

[5] Huansheng Ning, Hong Liu, and Laurence T Yang, "Cyberentity security in the internet of things," *Computer*, vol. 46, no. 4, pp. 46–53, 2013.

[6] Andrew Carlin, Mohammad Hammoudeh, and Omar Aldabbas, "Intrusion detection and countermeasure of virtual cloud systems-state of the art and current challenges," *INTERNATIONAL JOURNAL OF ADVANCED COMPUTER SCIENCE AND APPLICATIONS*, vol. 6, no. 6, pp. 1–15, 2015.

[7] Ibrahim Ghafir, Vaclav Prenosil, Ahmad Alhejailan, and Mohammad Hammoudeh, "Social engineering attack strategies and defence approaches," in *Future Internet of Things and Cloud (FiCloud), 2016 IEEE 4th International Conference on*. IEEE, 2016, pp. 145–149.

[8] Mohammad Hammoudeh, Fayez Al-Fayez, Huw Lloyd, Robert Newman, Bamidele Adebisi, Ahcène Bounceur, and Abdelrahman Abuarqoub, "A wireless sensor network border monitoring system: Deployment issues and routing protocols," *IEEE Sensors Journal*, 2017.

[9] Rodrigo Roman, Pablo Najera, and Javier Lopez, "Securing the internet of things," *Computer*, vol. 44, no. 9, pp. 51–58, 2011.

[10] Andrew Carlin, Mohammad Hammoudeh, and Omar Aldabbas, "Defence for distributed denial of service attacks in cloud computing," *Procedia Computer Science*, vol. 73, pp. 490–497, 2015.

[11] Nesrine Kaaniche, Maryline Laurent, and Mohammed El Barbori, "Cloudasec: A novel publickey based framework to handle data sharing security in clouds," in *11th IEEE International Conference on Security and Cryptography(Secrypt)*, 2014.

[12] Sana Belguith, Abderrazek Jemai, and Rabah Attia, "Enhancing data security in cloud computing using a lightweight cryptographic algorithm," in *ICAS 2015 : The Eleventh International Conference on Autonomic and Autonomous Systems*. 2015, pp. 98–103, IARIA.

[13] Nesrine Kaaniche, Aymen Boudguiga, and Maryline Laurent, "Id based cryptography for cloud data storage," in *2013 IEEE Sixth International Conference on Cloud Computing*. IEEE, 2013, pp. 375–382.

[14] Sabrina De Capitani Di Vimercati, Sara Foresti, Giovanni Livraga, and Pierangela Samarati, "Selective and private access to outsourced data centers," in *Handbook on Data Centers*, pp. 997–1027. Springer, 2015.

[15] Máté Horváth, "Attribute-based encryption optimized for cloud computing," in *SOFSEM 2015: Theory and Practice of Computer Science*, pp. 566–577. Springer, 2015.

[16] Joseph K Liu, Man Ho Au, Xinyi Huang, Rongxing Lu, and Jin Li, "Fine-grained two-factor access control for web-based cloud computing services," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 3, pp. 484–497, 2016.

[17] Nesrine Kaaniche and Maryline Laurent, "Attribute-based signatures for supporting anonymous certification," in *European Symposium on Research in Computer Security*. Springer, 2016, pp. 279–300.

[18] Lifeng Li, Xiaowan Chen, Hai Jiang, Zhongwen Li, and Kuan-Ching Li, "P-cp-abe: Parallelizing ciphertext-policy attribute-based encryption for clouds," in *Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2016 17th IEEE/ACIS International Conference on*. IEEE, 2016, pp. 575–580.

[19] Wei Li, Kaiping Xue, Yingjie Xue, and Jianan Hong, "Tmacs: A robust and verifiable threshold multi-authority access control system in public cloud storage," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 5, pp. 1484–1496, 2016.

[20] Tariq Alsboui, Mohammad Hammoudeh, and Abdelrahman Abuarqoub, "A service-centric stack for collaborative data sharing and processing," in *International Conferences on Green and Smart Technology with Sensor Applications2012 : GST and SIA*. 2012, pp. 320–327, Springer.

[21] Sana Belguith, Nesrine Kaaniche, Abderrazek Jemai, Maryline Laurent, and Rabah Attia, "Pabac: a privacy preserving attribute based framework for fine grained access control in clouds," in *13th IEEE International Conference on Security and Cryptography(Secrypt)*, 2016, pp. 133–146.

[22] Fuchun Guo, Yi Mu, Willy Susilo, Duncan S Wong, and Vijay Varadharajan, "Cp-abe with constant-size keys for lightweight devices," *IEEE transactions on information forensics and security*, vol. 9, no. 5, pp. 763–771, 2014.

[23] Nuttapong Attrapadung, Javier Herranz, Fabien Laguillaumie, Benoît Libert, Elie De Panafieu, and Carla Ràfols, "Attribute-based encryption schemes with constant-size ciphertexts," *Theoretical Computer Science*, vol. 422, pp. 15–38, 2012.

[24] Matthew Green, Susan Hohenberger, Brent Waters, et al., "Outsourcing the decryption of abe ciphertexts," in *USENIX Security Symposium*, 2011, number 3.

[25] Cong Zuo, Jun Shao, Guiyi Wei, Mande Xie, and Min Ji, "Cca-secure abe with outsourced decryption for fog computing," *Future Generation Computer Systems*, 2016.

[26] Takashi Nishide, Kazuki Yoneyama, and Kazuo Ohta, "Attribute-based encryption with partially hidden encryptor-specified access structures," in *International Conference on Applied Cryptography and Network Security*. Springer, 2008, pp. 111–129.

[27] Tran Viet Xuan Phuong, Guomin Yang, and Willy Susilo, "Hidden ciphertext policy attribute-based encryption under standard assumptions," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 1, pp. 35–45, 2016.

[28] Runhua Xu and Bo Lang, "A cp-abe scheme with hidden policy and its application in cloud computing," *International Journal of Cloud Computing*, vol. 4, no. 4, pp. 279–298, 2015.

[29] Allison Lewko and Brent Waters, "Decentralizing attribute-based encryption," in *Advances in Cryptology–EUROCRYPT 2011*, pp. 568–588. Springer, 2011.

[30] Melissa Chase and Sherman SM Chow, "Improving privacy and security in multi-authority attribute-based encryption," in *Proceedings of the 16th ACM conference on Computer and communications security*. ACM, 2009, pp. 121–130.

[31] "Health Insurance Portability and Accountability Act (HIPAA)," `https://www.hipaa.com/about/`, 2015.

[32] Amit Sahai and Brent Waters, Fuzzy identity-based encryption, "Fuzzy identity-based encryption," in *EUROCRYPT 2005*, pp. 457–473. Springer, 2005.

[33] John Bethencourt, Amit Sahai, and Brent Waters, "Ciphertext-policy attribute-based encryption," in *IEEE Symposium on Security and Privacy, 2007.*, 2007, pp. 321–334.

[34] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *The 13th ACM conference on Computer and communications security*, 2006, pp. 89–98.

[35] Brent Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Public Key Cryptography–PKC 2011*, pp. 53–70. Springer, 2011.

[36] Ling Cheung and Calvin Newport, "Provably secure ciphertext policy abe," in *Proceedings of the 14th ACM conference on Computer and communications security*. ACM, 2007, pp. 456–465.

[37] Vladimir Božović, Daniel Socek, Rainer Steinwandt, and Viktória I Villányi, "Multi-authority attribute-based encryption with honest-but-curious central authority," *International Journal of Computer Mathematics*, vol. 89, no. 3, pp. 268–283, 2012.

[38] Vanga Odelu, Ashok Kumar Das, Muhammad Khurram Khan, Kim-Kwang Raymond Choo, and Minho Jo, "Expressive cp-abe scheme for mobile devices in iot satisfying constant-size keys and ciphertexts," *IEEE Access*, vol. 5, pp. 3273–3283, 2017.

[39] Sana Belguith, Nesrine Kaaniche, Maryline Laurent, Abderrazak Jemai, and Rabah Attia, "Constant-size threshold attribute based signcryption for cloud applications," in *SECRYPT 2017: 14th International Conference on Security and Cryptography*, 2017, vol. 6, pp. 212–225.

[40] Vanga Odelu and Ashok Kumar Das, "Design of a new cp-abe with constant-size secret keys for lightweight devices using elliptic curve cryptography," *Security and Communication Networks*, vol. 9, no. 17, pp. 4048–4059, 2016.

[41] Vanga Odelu, Ashok Kumar Das, Y Sreenivasa Rao,

Saru Kumari, Muhammad Khurram Khan, and Kim-Kwang Raymond Choo, "Pairing-based cp-abe with constant-size ciphertexts and secret keys for cloud environment," *Computer Standards & Interfaces*, vol. 54, pp. 3–9, 2017.

[42] Yanjiang Yang, Haiyan Zhu, Haibing Lu, Jian Weng, Youcheng Zhang, and Kim-Kwang Raymond Choo, "Cloud based data sharing with fine-grained proxy re-encryption," *Pervasive and Mobile computing*, vol. 28, pp. 122–134, 2016.

[43] Xiaolong Xu, Jinglan Zhou, Xinheng Wang, and Yun Zhang, "Multi-authority proxy re-encryption based on cpabe for cloud storage systems," *Journal of Systems Engineering and Electronics*, vol. 27, no. 1, pp. 211–223, 2016.

[44] Ran Canetti and Susan Hohenberger, "Chosen-ciphertext secure proxy re-encryption," in *Proceedings of the 14th ACM conference on Computer and communications security*. ACM, 2007, pp. 185–194.

[45] Baodong Qin, Robert H Deng, Shengli Liu, and Siqi Ma, "Attribute-based encryption with efficient verifiable outsourced decryption," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 7, pp. 1384–1393, 2015.

[46] Kai-Min Chung, Yael Kalai, and Salil Vadhan, "Improved delegation of computation using fully homomorphic encryption," in *Annual Cryptology Conference*. Springer, 2010, pp. 483–501.

[47] Benoît Chevallier-Mames, Jean-Sébastien Coron, Noel McCullagh, David Naccache, and Michael Scott, "Secure delegation of elliptic-curve pairing," in *International Conference on Smart Card Research and Advanced Applications*. Springer, 2010, pp. 24–35.

[48] Rosario Gennaro, Craig Gentry, and Bryan Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in *Annual Cryptology Conference*. Springer, 2010, pp. 465–482.

[49] Craig Gentry et al., "Fully homomorphic encryption using ideal lattices.," in *STOC*, 2009, vol. 9, pp. 169–178.

[50] Junzuo Lai, Robert H Deng, Chaowen Guan, and Jian Weng, "Attribute-based encryption with verifiable outsourced decryption," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 8, pp. 1343–1354, 2013.

[51] Jin Li, Xinyi Huang, Jingwei Li, Xiaofeng Chen, and Yang Xiang, "Securely outsourcing attribute-based encryption with checkability," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 8, pp. 2201–2210, 2014.

[52] Jin Li, Xiaofeng Chen, Jingwei Li, Chunfu Jia, Jianfeng Ma, and Wenjing Lou, "Fine-grained access control system based on outsourced attribute-based encryption," in *European Symposium on Research in Computer Security*. Springer, 2013, pp. 592–609.

[53] Hao Wang, Bo Yang, and Yilei Wang, "Server aided ciphertext-policy attribute-based encryption," in *Advanced Information Networking and Applications Workshops (WAINA), 2015 IEEE 29th International Conference on*. IEEE, 2015, pp. 440–444.

[54] Suqing Lin, Rui Zhang, Hui Ma, and Mingsheng Wang, "Revisiting attribute-based encryption with verifiable outsourced decryption," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 10, pp. 2119–2130, 2015.

[55] Jiguo Li, Fengjie Sha, Yichen Zhang, Xinyi Huang, and Jian Shen, "Verifiable outsourced decryption of attribute-based encryption with constant ciphertext length," *Security and Communication Networks*, vol. 2017, 2017.

[56] Junzuo Lai, Robert H Deng, and Yingjiu Li, "Expressive cp-abe with partially hidden access structures," in *Proceedings of the 7th ACM symposium on information, computer and communications security*. ACM, 2012, pp. 18–19.

[57] Zhibin Zhou, Dijiang Huang, and Zhijie Wang, "Efficient privacy-preserving ciphertext-policy attribute based-encryption and broadcast encryption," *IEEE Transactions on Computers*, vol. 64, no. 1, pp. 126–138, 2015.

[58] Hong Zhong, Wenlong Zhu, Yan Xu, and Jie Cui, "Multi-authority attribute-based encryption access control scheme with policy hidden for cloud storage," *Soft Computing*, pp. 1–9, 2016.

[59] Ran Canetti, Hugo Krawczyk, and Jesper B Nielsen, "Relaxing chosen-ciphertext security," in *Annual International Cryptology Conference*. Springer, 2003, pp. 565–582.

[60] Aniket Kate, Greg Zaverucha, and Ian Goldberg, "Pairing-based onion routing," in *International Workshop on Privacy Enhancing Technologies*. Springer, 2007, pp. 95–112.

[61] Zhibin Zhou and Dijiang Huang, "Efficient and secure data storage operations for mobile cloud computing," in *Network and service management (cnsm), 2012 8th international conference and 2012 workshop on systems virtualiztion management (svm)*. IEEE, 2012, pp. 37–45.

[62] Moreno Ambrosin, Mauro Conti, and Tooska Dargahi, "On the feasibility of attribute-based encryption on smartphone devices," in *Proceedings of the 2015 Workshop on IoT challenges in Mobile and Industrial Sys-*

tems. ACM, 2015, pp. 49–54.

[63] Moreno Ambrosin, Arman Anzanpour, Mauro Conti, Tooska Dargahi, Sanaz Rahimi Moosavi, Amir M Rahmani, and Pasi Liljeberg, "On the feasibility of attribute-based encryption on internet of things devices," *IEEE Micro*, vol. 36, no. 6, pp. 25–35, 2016.

[64] Aleksandr Ometov, Pavel Masek, Lukas Malina, Roman Florea, Jiri Hosek, Sergey Andreev, Jan Hajny, Jussi Niutanen, and Yevgeni Koucheryavy, "Feasibility characterization of cryptographic primitives for constrained (wearable) iot devices," in *Pervasive Computing and Communication Workshops (PerCom Workshops), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1–6.

[65] Xinlei Wang, Jianqing Zhang, Eve M Schooler, and Mihaela Ion, "Performance evaluation of attribute-based encryption: Toward data privacy in the iot," in *Communications (ICC), 2014 IEEE International Conference on*. IEEE, 2014, pp. 725–730.