IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 13, 2018, accepted May 29, 2018, date of publication June 6, 2018, date of current version June 20, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2844373

Achieving Scalable Access Control Over Encrypted Data for Edge Computing Networks

HUI CUI[1,2], XUN YI[1], AND SURYA NEPAL[2]
[1]School of Science, RMIT University, Melbourne, VIC 3000, Australia
[2]Data61, CSIRO, Melbourne, VIC 3008, Australia

Corresponding author: Hui Cui (hui.cui@rmit.edu.au)

This work was supported by the Data61 Research Collaborative Project "Enhancing Security and Privacy in IoT."

ABSTRACT The concept of Internet of Things (IoT) has raised in the cloud computing paradigm as it adds latency when migrating all pieces of data from the network edge to the data center for them to be approached. Edge computing has been introduced to extend the cloud computing architecture to the edge of the network, which analyzes most of the IoT data near the devices that produce and act on that data. Though edge computing solves the latency problem of data processing, it also brings issues to the data security and privacy preservation. One technique which is potential to provide scalable access control to support data security and privacy in edge computing is attribute-based encryption (ABE). In this paper, we propose a primitive named proxy-aided ciphertext-policy ABE (PA-CPABE), which outsources the majority of the decryption computations to edge devices. Compared to the existing ABE with outsourced decryption schemes, PA-CPABE has an advantage in which the key distribution does not require any secure channels. We present a generic construction of PA-CPABE and then formally prove its security. In addition, we implement an instantiation of the proposed PA-CPABE framework to evaluate its performance.

INDEX TERMS Data security and privacy, access control, cloud computing, IoT security, edge computing.

I. INTRODUCTION

The idea of Internet of Things (IoT) has become increasingly popular, which enables various objects including physical devices, vehicles, buildings and other items embedded with computing and communication capabilities to exchange data. However, because of limitations in the computation capability, battery, storage and bandwidth, smart devices sometimes may decrease the quality of services and weaken the user experience. Cloud computing supplies resources to end users in terms of software, infrastructure and platform, and delivers services to applications at a comparatively small cost, which has been considered as a promising solution to mitigate the limitation of devices with constrained resources.

Unfortunately, cloud computing cannot be an answer to all emerging problems, since some IoT applications need to be instantly responded, some contain sensitive information, and some generate a large amount of data and cause a heavy workload to the network. The demand for distributing the IoT workloads between the local data center and the cloud has resulted in an architectural model called Edge Computing [1] (which is also known as Fog Computing [2]).

Edge computing extends cloud computing and facilitates cloud computing in significantly reducing the delays incurred by service deployments. End devices, edge and cloud form a three-layer hierarchical architecture (as shown in Fig. 1) for the service delivery, which supports a wide range of applications (e.g., the smart city network). Take the autonomous vehicle network as an instance, where the vehicle might produce gigabyte data in one second, and the real-time processing is in necessity as any delay in practice could lead the vehicle to make false resolutions [1]. In such a situation,

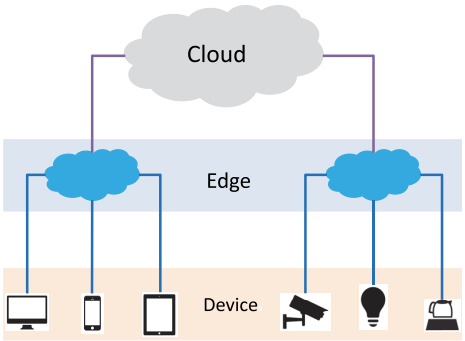


FIGURE 1. An architecture for Edge Computing.

the responding time could be extremely long if all data items are going to be forwarded to and processed by the cloud, and thus it would be very demanding for the current network to support a large number of vehicles in the same area. Therefore, it is essential for all kinds of data items to be managed at the network edge to reach a more effective management and a shorter response time.

Edge devices reduce communication and computation overheads by providing computing, networking and storage services and making decisions at the network edge. Unfortunately, edge devices requiring less cost than cloud servers can be easily compromised by adversaries and cannot be trusted, especially in the data sharing (e.g., vehicles may need to share the traffic data when traveling on the same motorway) situation. Therefore, it is indispensable to arm an edge computing network with an access control mechanism to allow the data to be shared among data users possessing certain attributes while preventing other entities (including the cloud server, edge devices and unprivileged data users) from learning the original data.

Attribute-based encryption (ABE) [3] protects data security and privacy by sharing data among a group of privileged data users, which is believe to be a very desirable candidate for accomplishing scalable (i.e., fine-grained) access control over data items in encrypted forms. One feature of current ABE schemes is that they are built upon bilinear pairings (or bilinear maps), and thus it is significantly challenging to deploy such schemes in applications where the private data will be accessed via a mobile device with a constrained computation capacity. With this issue in mind, Green *et al.* [4] suggested to divide the private attribute-key in an ABE scheme into a transformation key and a decryption key, of which the former is sent to a proxy such that the proxy can make a transformation on the ciphertext (to produce a partially decrypted ciphertext) and the latter is given to the data user such that the data user can completely decrypt the transformed ciphertext. Following this direction of delegating the workloads in the decryption to a third party like a proxy, in terms of enhancing data security and privacy to meet different requirements in the real world, several ABE schemes enabling outsourced decryption (e.g., [5], [6]) have been proposed.

ABE with outsourced decryption (ABE-OD) has an inherent property to be implemented in an edge computing network to enforce access control and protect data security and privacy, where the edge device can play the role of the proxy. However, all existing ABE-OD schemes require secure channels to distribute private keys to data users, which is not feasible for all applications in the edge computing network due to the expensive cost in building secure channels. Motivated by this observation, we consider designing a secure channel free ABE-OD scheme (to distinguish from ABE-OD, we call it proxy-aided ciphertext-policy ABE (PA-CPABE)) to provide scalable access control over data items in encrypted forms in the edge computing network. Our aim is to give a generic transformation technique which is able to convert any ciphertext-policy attribute-based encryption (CP-ABE) [7] scheme into a PA-CPABE scheme. Briefly speaking, the contributions in this paper are threefold.

- We put forth a primitive called proxy-aided ciphertext-policy attribute-based encryption (PA-CPABE) to outsource the decryption workloads of ABE ciphertexts to an untrusted proxy (i.e., an edge device) but without requiring any secure channels for the key distribution, which can be seamlessly integrated into the edge computing network to accomplish the scalable access control.
- We give a generic construction for PA-CPABE via which a PA-CPABE scheme could be converted from a CP-ABE scheme, and then apply a concrete CP-ABE scheme which satisfies certain properties into the generic construction of PA-CPABE to obtain a concrete PA-CPABE scheme.
- We implement the proposed concrete PA-CPABE scheme as well as its underlying CP-ABE scheme to assess the practicability of the former and show that PA-CPABE significantly ameliorates the decryption cost incurred for the data user in an ordinary CP-ABE scheme.

A. RELATED WORK

To facilitate the deployment of cloud computing and internet of things (IoT) services, edge computing [1] allows to conduct computation on the data at the network edge. Despite the advantages of edge computing, many challenges have raised as well, including data abstraction, programmability, data security and privacy, service management and optimization metrics, el al. [1]. In this paper, the focus is the preservation of data security and privacy for edge computing.

Thanks to the property of enabling access control over data items that are encrypted, a primitive to preserve data security and privacy for scenarios like cloud computing called attribute-based encryption (ABE) [3] has been intensively adopted in cloud relevant applications since its introduction in 2005. However, ABE has several drawbacks which impede its usability in the real world, especially its expensiveness in the computation which makes it impractical for resourced constrained devices to run ABE related algorithms. Existing ABE schemes (e.g., [7], [8]) are built from bilinear pairings, and thus their decryption algorithms require expensive pairing operations (one pairing operation usually takes three times more than one exponentiation operation). To address this problem, Green *et al.* [4] recommended to outsource the decryption workload in ABE to a proxy (or a server) where the private attribute-key is divided into a transformation key for the proxy and a decryption key for the data user such that only one exponentiation operation is needed to be conducted by the data user to decrypt the result received from the proxy to obtain the original message. The proxy is not a trusted entity, so it may not do the calculation in a

correct way. To address this issue in ABE with outsourced decryption (ABE-OD), Lai *et al.* [5] proposed an a construction on ABE with verifiable and outsourced decryption (ABE-VOD), but that scheme adds significant amount of calculations to the original ABE scheme and thus is not efficient. Li *et al.* [6] suggested to check whether the result of the outsourced decryption in an ABE-VOD scheme is correct in a distributed manner to improve the efficiency, but more than one key generation center (KGC) are assigned in the ABE-VOD scheme and at least one of them should be honest and take the correct ciphertext as the input. Qin *et al.* [9] and Mao *et al.* [10] presented generic constructions on ABE-VOD, respectively, which can transform any ABE-OD scheme to an ABE-VOD scheme. Fan *et al.* [11] presented a revocable ABE-VOD scheme in the setting of multiple KGCs where the role of the single key generation center (KGC) is split across multiple KGCs.

B. ROADMAP

The rest of this paper is going to be structured in the following way. In Section II, the notations and notions relevant to this paper are revisited. In Section III, the system framework and the security definition for a proxy-aided ciphertext-policy attribute-based encryption (PA-CPABE) scheme are described. In Section IV, a generic construction on PA-CPABE, and an instantiation of PA-CPABE are presented. In Section V, in addition to the comparison result between PA-CPABE and other related works, the implementation result of the proposed instantiation is detailed. Finally, this paper is concluded in Section VI.

II. PRELIMINARIES

In this section, we briefly delineate several notations and terminologies that are going to be utilized in this paper.

A. ACCESS STRUCTURES AND SECRET SHARING SCHEMES

Informally speaking, authorized sets consist of parties in groups that are given access, and an access structure is the set of all such authorized sets, which describes that who should be work with whom in order to have access to a resource (i.e., secret). A scheme where the secret is shared by different parties and only those subgroups of parties included in the access structure are capable of recomputing the secret by putting their shares together is called a (linear) secret sharing scheme. If a subset, say S, belongs to an access structure, and all sets containing the subset S are covered by this access structure as well, then this access structure is said to be monotone.

Below the formal notions for an access structure, as well as a secret sharing scheme, are described.

Definition 1 (Access Structures [8], [12], [13]): Take a collection of parties $P = \{P_1, ..., P_n\}$ into consideration. A set $\mathbb{A} \subseteq 2^P$ is said to be monotone if for all B and C, $C \in \mathbb{A}$ holds when $B \in \mathbb{A}$ and $B \subseteq C$. In essence, an access structure is composed of a class $\mathbb{A}$ of non-empty subsets of the parties P, i.e., $\mathbb{A} \subseteq 2^P \setminus \{\emptyset\}$. Any set in the access structure $\mathbb{A}$ is defined to be an authorized set, and any set not in the access structure $\mathbb{A}$ is defined to be an unauthorized set.

Definition 2 (Linear Secret Sharing Schemes [8], [12], [14]): Consider P as a class of entities. Denote $\mathbb{M}$ as a matrix having n columns and l rows, and $\rho : \{1, ..., l\} \rightarrow P$ as a function mapping a row to an entity for the labeling purpose (note that the pair $(\mathbb{M}, \rho)$ will also be referred to as the access structure $\mathbb{A}$ in this paper). A secret sharing scheme Π over a set of entities P satisfying the following properties is said to be a linear secret sharing (LSS) scheme over Z_p.

1) A vector over Z_p can be formed from the shares of each party.
2) There exists a share-generating matrix $\mathbb{M}$ with l rows and n columns in association with the secret sharing scheme Π. Suppose that for any $i \in [1, l]$, an entity $\rho(i)$ is expressed to label the i-th row of the matrix $\mathbb{M}$. Assume that $\overrightarrow{v} = (\mu, r_2, ..., r_n)$ is a column vector with $\mu \in Z_p$ being the secret which is going to be shared and $r_2, ..., r_n \in Z_p$ being randomly chosen elements. Then $\mathbb{M}\overrightarrow{v}$ is the vector of l shares of the secret μ in terms of the secret sharing scheme Π. Thus, an entity $\rho(i)$ actually possesses a share $(\mathbb{M}\overrightarrow{v})_i$.

It has been stated in [12] that each LSS scheme is equipped with a property known as linear reconstruction. Let Π be an LSS scheme for an access structure $\mathbb{A}$ with an authorized set A. Define $I \subseteq \{1, ..., l\}$ as $I = \{i | \rho(i) \in A\}$. Then the span of rows of the matrix $\mathbb{M}$ which is indexed by I includes a vector $(1, 0, ..., 0)$, and it is not difficult to find constants $\{w_i \in Z_p\}_{i \in I}$ (regarding the size of the share-generating matrix $\mathbb{M}$) satisfying $\sum_{i \in I} w_i v_i = \mu$ for any valid shares $\{v_i\}$ of a secret μ in terms of the LSS scheme Π in the polynomial time [15].

B. CIPHERTEXT-POLICY ATTRIBUTE-BASED ENCRYPTION

Denote $\mathbf{CAE} = (\text{CAE.Setup, CAE.KeyGen, CAE.Encrypt, CAE.Decrypt})$ as a ciphertext-policy attribute-based encryption (CP-ABE) (e.g., [16]), where CAE.Setup is a setup algorithm which creates the public parameter pm_A as well as the master private key mk_A on input the security parameter λ, CAE.KeyGen is a private attribute-key generation algorithm which creates a private attribute-key sk_A for an attribute set A on input the public parameter pm_A, the master private key mk_A and a set of attributes A, CAE.Encrypt is an encryption algorithm which creates a ciphertext CT associated with an access structure structure $\mathbb{A}$ on input the public parameter pm_A, an access structure $\mathbb{A}$ and a message M, and CAE.Decrypt is a decryption algorithm which creates a message M if the attributes A of a private attribute-key sk_A is an authorized set of the access structure $\mathbb{A}$ ascribed to a ciphertext CT or a failure symbol $\perp$ otherwise on input the public parameter pm_A, a ciphertext CT associated with an access structure $\mathbb{A}$ and a private attribute-key sk_A over attributes A [17].

When a CP-ABE scheme **CAE** is considered to be correct, it means that for any security parameter $\lambda \in \mathbb{N}$, any message M (in the message space), any authorized attribute set A (in the space of attributes) for any access structure $\mathbb{A}$ (in the space of access structures), if $(pm_A, mk_A) \leftarrow \text{CAE.Setup}(1^\lambda)$, $sk_A \leftarrow \text{CAE.KeyGen}(pm_A, mk_A, A)$, $\text{CT} \leftarrow \text{CAE.Encrypt}(pm_A, \mathbb{A}, M)$, it holds that $\text{CAE.Decrypt}(pm_A, \text{CT}, sk_A) = M$.

Let m_0 and m_1 be two messages of the same length, and $\mathcal{O}_{\text{KG}(\cdot)}$ be the private attribute-key generation oracle which outputs a private attribute-key sk_A associated with a set of attributes A by taking the public parameter pm_A, the master private key mk_A and a set of attributes A as the input with the restriction that any set of attributes A satisfying the challenge access structure $\mathbb{A}^*$ is disallowed to be queried to the $\mathcal{O}_{\text{KG}(\cdot)}$ oracle. Regarding any probabilistic polynomial time (PPT) adversary $\mathcal{A}$, if the advantage function

$$Adv_{CAE,\mathcal{A}}^{\text{IND-CPA}}(\lambda) = \Pr\left[b' = b \;\middle|\; \begin{array}{l} (pm_A, mk_A) \leftarrow \text{CAE.Setup}(1^\lambda), \quad b \leftarrow \{0,1\} \\ (m_0, m_1, \mathbb{A}^*, state) \leftarrow \mathcal{A}^{\mathcal{O}_{\text{KG}(\cdot)}}(pm_A) \\ \text{CT}^* \leftarrow \text{CAE.Encrypt}(pm_A, \mathbb{A}^*, m_b) \\ b' \leftarrow \mathcal{A}^{\mathcal{O}_{\text{KG}(\cdot)}}(pm_A, m_0, m_1, \mathbb{A}^*, state, \text{CT}^*) \end{array}\right] - 1/2$$

is negligible in the security parameter λ, then a CP-ABE scheme **CAE** is regarded to be indistinguishable under chosen plaintext attacks (shortly, IND-CPA secure). In addition, if there exists an Init phase before the CAE.Setup phase which gives the challenge access structure $\mathbb{A}^*$ the adversary $\mathcal{A}$ aims to attack, then a CP-ABE scheme **CAE** is regarded to be selectively IND-CPA secure.

C. PUBLIC-KEY ENCRYPTION

Let **PE** = (PE.Setup, PE.KeyGen, PE.Encrypt, PE.Decrypt) denote a public-key encryption (PKE) scheme (e.g., [18]) where PE.Setup is a setup algorithm which generates the public parameter pm_P by taking a security parameter λ as the input, PE.KeyGen is a key generation algorithm which generates a public and private key pair (pk_{id}, sk_{id}) for the data user id by taking the public parameter pm_P and a data user id as the input, PE.Encrypt is an encryption algorithm which generates a ciphertext CT by taking the public parameter pm_P, a public key pk_{id} and a message M as the input, and PE.Decrypt is a decryption algorithm which generates a message M or a failure symbol $\perp$ by taking the public parameter pm_P, a ciphertext and a private key sk_{id} of a data user id as the input [19].

When a PKE scheme **PE** is considered to be correct, it means that for any security parameter $\lambda \in \mathbb{A}$, any message M (in the message space), if $pm_P \leftarrow \text{PE.Setup}(1^\lambda)$, $(pk_{id}, sk_{id}) \leftarrow \text{PE.KeyGen}(pm_P, id)$, $\text{CT} \leftarrow \text{PE.Encrypt}(pm_P, pk_{id}, M)$, it holds that $\text{PE.Decrypt}(pm_P, \text{CT}, sk_{id}) = M$.

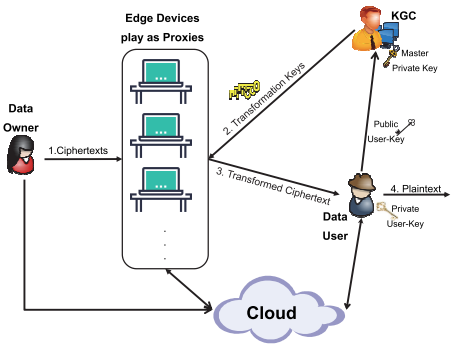


FIGURE 2. A pictorial system architecture of PA-CPABE.

Let m_0 and m_1 be two messages of the same size. Regarding any PPT adversary $\mathcal{A}$, if the advantage function

$$Adv_{PE,\mathcal{A}}^{\text{IND-CPA}}(\lambda) = \Pr\left[b' = b \;\middle|\; \begin{array}{l} pm_P \leftarrow \text{PE.Setup}(1^\lambda), \quad b \leftarrow \{0,1\} \\ (pk_{id}, sk_{id}) \leftarrow \text{PE.KeyGen}(pm_P, id) \\ (m_0, m_1, state) \leftarrow \mathcal{A}(pm_P, pk_{id}) \\ \text{CT}^* \leftarrow \text{PE.Encrypt}(pm_P, pk_{id}, m_b) \\ b' \leftarrow \mathcal{A}(pm_P, pk_{id}, m_0, m_1, state, \text{CT}^*) \end{array}\right] - 1/2$$

is negligible in the security parameter λ, a PKE scheme **PE** is regarded to be IND-CPA secure (i.e., indistinguishable under chosen plaintext attacks).

III. FRAMEWORK AND SECURITY DEFINITION

We define the framework, as well as the security model, for proxy-aided ciphertext-policy attribute-based encryption (PA-CPABE), in this section.

A. SYSTEM OVERVIEW

The architecture of a PA-CPABE scheme is depicted under the scenario of an edge computing network in Fig. 2, involving four entities: data users, the trusted key generation center (KGC), untrusted proxies (i.e., edge nodes or edge devices) and data owners. The KGC is in charge of the creation of the common parameter and the master private key. The KGC keeps the latter in secret and make the former public. Once a data user Bob intends to join the network, he registers with the KGC. Firstly, Bob creates a public user-key and a private user-key. Then, Bob transmits the public user-key to the KGC (along with a proof about his knowledge of the corresponding private user-key) and keeps the private user-key as a secret. The KGC, on the basis of Bob's eligible attributes and public user-key, produces a public transformation key for Bob, which is going to be broadcast to all local edge nodes of Bob. Before uploading a message (e.g., a document or a file) to the cloud, a data owner Alice uses the common public parameter to encrypt the message over an access structure she specifies. The resulting ciphertext for the message (rather than the plaintext of the message) is sent to the nearby edge device which will forward the ciphertext to the cloud if necessary.

In case that Bob needs to access a ciphertext, Bob transmits to the cloud a request, and the cloud will forward the ciphertext to the nearby edge device which is capable of performing the computation. If the attribute set possessed by Bob satisfies (i.e., is an authorized set of attribute for) the access structure associated with the ciphertext, the edge device is capable of using the transformation key of Bob to partially decrypt (i.e., transform) the ciphertext. After obtaining the transformed ciphertext from the edge device, Bob uses his private user-key to fully decrypt it to obtain the underlying plaintext.

Notice that all data users in PA-CPABE only need to communicate with the KGC when they firstly register with an edge computing network. In addition, the operations handled by the untrusted edge devices (i.e., the proxies) are completely transparent to data users. In other words, a data user, if necessary, can check whether the partial decryption has been correctly conducted via transforming the ciphertext by himself/herself.

B. FRAMEWORK

The algorithms for a proxy-aided ciphertext-policy attribute-based encryption (PA-CPABE) scheme **PCAE** are as follows, of which some are similar to those in [13].

- $\text{Setup}(1^\lambda) \rightarrow (pm, mk)$. This algorithm is run by the key generation center (KGC). It takes the security parameter λ as the input, and outputs the public parameter pm and the master private key mk.
- $\text{UserKG}(pm, id) \rightarrow (pk_{id}, sk_{id})$. This algorithm is run by each data user id. It takes the public parameter pm and a data user id as the input, and outputs a public user-key pk_{id} and a private user-key sk_{id} for the data user id.
- $\text{PubKG}(pm, mk, pk_{id}, A) \rightarrow pk_{id}^A$. This algorithm is run by the KGC. It takes the public parameter pm, the master private key mk, a public user-key pk_{id} and a set of attributes A of a data user id as the input, and outputs a public transformation key pk_{id}^A for the data user id.
- $\text{Encrypt}(pm, \mathbb{A}, M) \rightarrow \text{CT}$. This algorithm is run by the data owner. It takes the public parameter pm, an access structure $\mathbb{A}$ and a message M (in the space of messages) as the input, and outputs a ciphertext CT.
- $\text{Transform}(pm, \text{CT}, pk_{id}^A) \rightarrow \text{CT}'$. This algorithm is run by the proxy (or an edge device). It takes the public parameter pm, a ciphertext CT for an access structure $\mathbb{A}$ and a public transformation key pk_{id}^A for an attribute set A of a data user id as the input, and outputs the transformed (or partially decrypted) ciphertext CT' if the attribute set A is a set of authorized attributes for the access structure $\mathbb{A}$.
- $\text{Decrypt}(pm, \text{CT}', sk_{id}) \rightarrow M/\perp$. This algorithm is run by the data user id. It takes the public parameter pm, a partially decrypted (or transformed) ciphertext CT' and a private user-key sk_{id} of a data user id as the input, and outputs a message M or a failure symbol $\perp$.

We say that a PA-CPABE scheme **PCAE** is correct, meaning that for any security parameter $\lambda \in \mathbb{N}$, any message M (in the message space), any data user id's set of attributes A (in the space of attributes) satisfies any access structure $\mathbb{A}$ (in the space of access structures), if $(pm, mk) \leftarrow \text{Setup}(1^\lambda)$, $(pk_{id}, sk_{id}) \leftarrow \text{UserKG}(pm, id)$, $pk_{id}^A \leftarrow \text{PubKG}(pm, mk, pk_{id}, A)$, $\text{CT} \leftarrow \text{Encrypt}(pm, \mathbb{A}, M)$, $\text{CT}' \leftarrow \text{Transform}(pm, \text{CT}, pk_{id}^A)$, we have $\text{Decrypt}(pm, \text{CT}', sk_{id}) = M$.

C. ADVERSARIAL MODEL

Considering the adversarial model for PA-CPABE, the proxy (i.e., an edge node or an edge device) is assumed to be untrusted such that the proxy may be in collusion with data users but the proxy itself does not hold any secret such that anybody is able to execute all operations conducted by the proxy (which implies that any misbehavior of the proxy can be conveniently observed), and the key generation center (KGC) is assumed to be trusted (implying that the master private key is always secretly kept and should not be leaked anyway). Thus, the adversary is given access to private user-keys, as well as data users' public transformation keys and attribute sets of its choice (except those data users with attributes satisfying the challenge access structure), but the adversary can never acquire any information related to the original plaintext hidden in a ciphertext in association with the challenge access structure.

Below the indistinguishability under chosen plaintext attacks (shortly speaking, the IND-CPA security) is defined between a challenger algorithm $\mathcal{B}$ and an adversary algorithm $\mathcal{A}$ for a PA-CPABE scheme **PCAE**.

- Setup Phase. For the generation of the public parameter pm and the master private key mk, algorithm $\mathcal{B}$ runs the $\text{Setup}(1^\lambda)$ algorithm. Algorithm $\mathcal{B}$ keeps the master private key mk in secret, and sends to algorithm $\mathcal{A}$ the public parameter pm. In addition, algorithm $\mathcal{A}$ creates a list L which is initially empty to store $(id, (pk_{id}, sk_{id}))$ for data users.
- Phase 1. The following queries are adaptively issued to algorithm $\mathcal{B}$ by algorithm $\mathcal{A}$.
  - Private-User-Key oracle. For any private user-key query on a data user id issued by algorithm $\mathcal{A}$, in order to return a private user-key sk_{id}, algorithm $\mathcal{B}$ runs the $\text{UserKG}(pm, id)$ algorithm. Notice that algorithm $\mathcal{B}$ adds (id, pk_{id}, sk_{id}) to a list L whenever it runs the $\text{UserKG}(pm, id)$ algorithm so that the same key pair (pk_{id}, sk_{id}) will be used for all queries on the same data user id.
  - Transformation-Key oracle. For any public transformation-key query on a data user id and a set of attributes A issued by algorithm $\mathcal{A}$, in order to return a transformation key pk_{id}^A, algorithm $\mathcal{B}$ runs the $\text{UserKG}(pm, id)$ algorithm (if no private user-key query on a data user id has been issued to the Private-User-Key oracle) and the $\text{PubKG}(pm, mk, pk_{id}, A)$ algorithm.
- Challenge Phase. Algorithm $\mathcal{A}$ outputs two messages m_0^*, m_1^* (m_0^* and m_1^* are of the same size), an access

structure $\mathbb{A}^*$ with the constraint that a query on (id^*, A^*) satisfying the challenge access structure $\mathbb{A}^*$ should never be issued to the Transformation-Key oracle once a private user-key query on a data user id^* has been issued to the Private-User-Key oracle. In order to respond to algorithm $\mathcal{A}$, algorithm $\mathcal{B}$ randomly chooses $b \in \{0, 1\}$, and runs the Encrypt(pm, $\mathbb{A}^*$, m_b^*) algorithm to generate the challenge ciphertext CT*.

- Phase 2. Following that restriction declared in the Challenge stage, algorithm $\mathcal{A}$ continues querying to the Private-User-Key and Transformation-Key oracles as in Phase 1.
- Guess Phase. Algorithm $\mathcal{A}$ outputs a guess b' for b. Algorithm $\mathcal{A}$ wins when $b' = b$.

The advantage of algorithm $\mathcal{A}$ in the IND-CPA security game for a PA-CPABE scheme **PCAE** is defined to be $\Pr[b = b'] - 1/2$. The PA-CPABE scheme **PCAE** is considered to be IND-CPA secure if a PPT adversary has at most a negligible advantage in the security parameter λ. Note that when there is an Init phase before the Setup stage, during which algorithm $\mathcal{A}$ outputs the challenge access structure $\mathbb{A}^*$ that it targets to attack, the PA-CPABE scheme **PCAE** is considered to be selectively IND-CPA secure.

IV. CONSTRUCTIONS ON PROXY-AIDED CIPHERTEXT-POLICY ATTRIBUTE-BASED ENCRYPTION

A generic construction and an instantiation for proxy-aided ciphertext-policy attribute-based encryption (PA-CPABE), as well as their security analysis, are illustrated in detail in this section.

A. INTUITION

The key challenge of building a PA-CPABE scheme is a generic key splitting technique which is able to divide the private attribute-key in a CP-ABE scheme into two parts, of which one is set as the public transformation key and sent to the proxies (i.e., edge nodes or edge devices) and the other one which is kept and generated by the data user himself/herself is defined as the private user-key. It is worthwhile to notice that neither a public transformation key nor the private user-key can be individually input to decrypt a ciphertext, which implies that the public transformation key and the private user-key have to work together to obtain the plaintext of a ciphertext. With the goal of binding the private user-key generated by each data user himself/herself[1] to the public transformation key of this data user generated by the KGC,[2] we make use of the key regeneration (also know as delegate) property which is inherent with a standard CP-ABE scheme. Interestingly, we found that such a delegate property can be subtly utilized to embed the public user-key (which has a corresponding private user-key and is created by a data user himself/herself) into the public transformation key (which is used by the proxy to convert a ciphertext into a partially decrypted ciphertext) and leave the corresponding private user-key (which works as a trapdoor) for the data user to fully decrypt a (partially decrypted) ciphertext.

[1]This removes the need of secure channels between the key generation center (KGC) and all data users for the secure delivery of the private user-key.

[2]Note that the KGC will authenticate the identity or public user-key of a data user and his/her eligible attributes before issuing the corresponding public transformation key.

B. GENERIC CONSTRUCTION

In general, a standard ciphertext-policy attribute-based encryption (CP-ABE) scheme is born with a property known as Delegate [7]. Specifically, assume that pm_A is the public parameter and mk_A is the master private key created by the setup algorithm CAE.Setup of a CP-ABE scheme **CAE**. Given a private attribute-key sk_A over an attribute set A which is generated by running CAE.KeyGen(pm_A, mk_A, A; r) where r is the chosen randomness, it is easy to regenerate a delegated key $sk'_{A'}$ over an attribute set $A' \subseteq A$ by running Delegate(pm_A, sk_A, A'; r') with r' being the chosen randomness, which is equivalent to a private attribute-key $sk_{A'}$ over an attribute set A' generated by the KGC running CAE.KeyGen(pm_A, mk_A, A'; $r \circ r'$) where $\circ$ is an operation such as "$\times$" and "$+$". In other words, the key created by Delegate(pm_A, sk_A, A'; r') is indistinguishable to the one generated by CAE.KeyGen(pm_A, mk_A, A'; $r \circ r'$). We observe that this property can be extended one step further which we call Extended Delegate (ExDelegate) satisfying the following definitions.

- The key generated by the ExDelegate(pm_A, sk_A, A'; r') algorithm is equivalent to the one created by the CAE.KeyGen(pm_A, $mk_A \circ r'$, A'; r) algorithm.
- The following two distributions upon a set of attributes A

$$\left\{ \begin{array}{l} (pk', r') \leftarrow \text{PE.KeyGen}(pm_P, id); \\ sk'_A \leftarrow \text{CAE.KeyGen}(pm_A, mk_A \circ r', A); \end{array} : (pk', sk'_A) \right\},$$

and

$$\left\{ \begin{array}{l} (pk', r') \leftarrow \text{PE.KeyGen}(pm_P, id); \\ \widetilde{sk}'_A \leftarrow \text{CAE.KeyGen}(pm_A, mk_A \circ r^*, A); \end{array} : (pk', \widetilde{sk}'_A) \right\}$$

are computationally indistinguishable where PE.KeyGen is the key generation algorithm of a public-key encryption (PKE) scheme PE which is deterministic such that the public key pk' is deterministically computed from the private key r' (this can be guaranteed by the IND-CPA security of a PKE scheme).

Take the CP-ABE scheme in [7] as an example, where the private attribute-key is $sk_A = (f^{\alpha+r}, \{g^r \cdot H(j)^{r_j}, g^{r_j}\}_{j \in A})$ with α being the master private key, group elements g, f and the hash function H belonging to the public parameter. Below we show that it satisfies the Extended Delegate property. Given CAE.KeyGen(pm_A, α, A; $(r, \{r_j\}_{j \in A})$) $= sk_A$ $= (f^{\alpha+r}, \{g^r \cdot H(j)^{r_j}, g^{r_j}\}_{j \in A})$, it is easy to compute a key as ExDelegate(pm_A, sk_A, A'; r') $= sk'_{A'} = ((f^{\alpha+r})^{r'}, \{(g^r \cdot H(j)^{r_j})^{r'}, (g^{r_j})^{r'}\}_{j \in A'}) = (f^{\alpha \cdot r' + r \cdot r'}, \{g^{r \cdot r'} \cdot H(j)^{r_j \cdot r'}, g^{r_j \cdot r'}\}_{j \in A'})$ $=$ CAE.KeyGen(pm_A, $\alpha \cdot r'$, A'; $(r \cdot r', \{r' \cdot r_j\}_{j \in A'})$) $=$ CAE.KeyGen($pm_A, \alpha \cdot r', A'; (r, \{r_j\}_{j \in A'})$) (setting $r = r \cdot r'$, r_j

$= r_j \cdot r'$). In addition, for a public user-key $pk' = g^{r'}$ and a private user-key $sk' = r'$ generated by the PE.KeyGen algorithm, it is not difficult to see that $(g^{r'}, (f^{\alpha \cdot r'} + r, \{g^r \cdot H(j)^{r_j}, g^{r_j}\}_{i \in A'}))$ and $(g^{r'}, (f^{\alpha \cdot r^* + r}, \{g^r \cdot H(j)^{r_j}, g^{r_j}\}_{i \in A'}))$ are computationally indistinguishable.

Denote **PE** = (PE.Setup, PE.KeyGen, PE.Encrypt, PE.Decrypt) as an IND-CPA secure PKE scheme with a deterministic PE.KeyGen algorithm, and **CAE** = (CAE.Setup, CAE.KeyGen, CAE.Encrypt, CAE.Decrypt) as an IND-CPA secure CP-ABE scheme with the ExDelegate property. A generic construction of PA-CPABE, which is composed of six algorithms, is given as follows.

- Setup(1^λ). On input the security parameter λ, this algorithm runs $(pm_A, mk_A) \leftarrow$ CAE.Setup(1^λ), and $pm_P \leftarrow$ PE.Setup(1^λ). It outputs $mk = mk_A$ as the master private key and $pm = (pm_P, pm_A)$ as the public parameter.
- UserKG(pm, id). On input the public parameter pm and a data user id, this algorithm runs $(pk_{id}, sk_{id}) \leftarrow$ PE.KeyGen(pm, id), and outputs pk_{id} as a public user-key and sk_{id} as a private user-key for the data user id.
- PubKG(pm, mk, pk_{id}, A). On input the public parameter pm, the master private key mk, a user id who has a public user-key pk_{id} and a set of attributes A, this algorithm runs $pk_{id}^A \leftarrow$ CAE.KeyGen(pm_A, $mk_A \circ pk_{id}$, A), where $\circ$ is a group operation. It outputs a transformation key pk_{id}^A for the data user id.
- Encrypt(pm, $\mathbb{A}$, M). On input the public parameter pm, an access structure $\mathbb{A}$ and a message M (in the message space), this algorithm runs CT $\leftarrow$ CAE.Encrypt(pm_A, $\mathbb{A}$, M). It outputs a ciphertext CT.
- Transform(pm, CT, pk_{id}^A). On input the public parameter pm, a ciphertext CT and a public transformation key pk_{id}^A for a data user id with attributes A, this algorithm runs CT$' \leftarrow$ CAE.Decrypt(pm_A, CT, pk_{id}^A). It outputs CT$'$ the transformed ciphertext.
- Decrypt(pm, CT$'$, sk_{id}). On input the public parameter pm, a transformed ciphertext CT$'$ and a data user id's private user-key sk_{id}, this algorithm runs $M \leftarrow$ PE.Decrypt(pm_P, CT$'$, sk_{id}). It outputs the plaintext M for a successful decryption and $\perp$ otherwise.

For the correctness of a PA-CPABE scheme, we require the underlying CP-ABE scheme **CAE** to be transformable to a PKE scheme **PE** such that a ciphertext generated by the encryption algorithm in a CP-ABE scheme **CAE** should be able to be transformed to a ciphertext for the same message created by the encryption algorithm in a PKE scheme **PE**. At a high level, for any data user id with an authorized attribute set A for an access structure $\mathbb{A}$ and a public user-key pk_{id} and a private user-key sk_{id} generated by the key generation algorithm in a PKE scheme **PE**, it follows that CAE.Decrypt(pm_A, CAE.Encrypt(pm_A, $\mathbb{A}$, M), CAE.KeyGen(pm_A, $mk_A \circ pk_{id}$, A)) = PE.Encrypt(pm_P, pk_{id}, M) = CT$'$ such that PE.Decrypt(pm_P, CT$'$, sk_{id}) = M.

Theorem 1: The proposed generic construction on PA-CPABE is (selectively) IND-CPA secure under the assumption that the CP-ABE scheme **CAE** which satisfies the Extended Delegate property is (selectively) IND-CPA secure, and the PKE scheme **PE** is IND-CPA secure.

Proof: If there is an adversary algorithm $\mathcal{A}$ which is able to break the IND-CPA security of the PA-CPABE scheme, then we are able to construct an adversary algorithm $\mathcal{A}_0$ which is able to break the IND-CPA security of the CP-ABE scheme **CAE** or the PKE scheme **PE**. Let $\mathcal{B}_0$ denote the challenger algorithm for the CP-ABE scheme **CAE** and $\mathcal{B}_1$ denote the challenger algorithm for the PKE scheme **PE**. Note that for the selective IND-CPA security, an Init phase during which a challenge access structure $\mathbb{A}^*$ is outputted by algorithm $\mathcal{A}$ is going to be defined before the Setup stage, which algorithm $\mathcal{A}_0$ sets as its own output in the Init stage for the selective IND-CPA security game of the underlying CP-ABE scheme **CAE**.

- Setup Phase. Algorithm $\mathcal{A}_0$ is given pm_A from algorithm $\mathcal{B}_0$ of the CP-ABE scheme **CAE**, and pm_P, pk^* from algorithm $\mathcal{B}_1$ of the PKE scheme **PE**. Algorithm $\mathcal{A}_0$ sends $pm = (pm_P, pm_A)$ to algorithm $\mathcal{A}$, and keeps a list L storing $(id, (pk_{id}, sk_{id}))$ for data users which is initially empty.
- Phase 1. The following queries to algorithm $\mathcal{A}_0$ are adaptively issued by algorithm $\mathcal{A}$.
  - Private-User-Key oracle on a data user id. When receiving a private user-key query on a data user id from algorithm $\mathcal{A}$, algorithm $\mathcal{A}_0$ returns a private user-key sk_{id} by running the UserKG algorithm. Algorithm $\mathcal{A}_0$ adds $(id, (pk_{id}, sk_{id}))$ to the list L such that the same (pk_{id}, sk_{id}) is going to be used for all queries on the same data user id.
  - Transformation-Key oracle on a set of attributes A and a data user id. For a transformation-key query on a data user id with a set of attributes A issued by algorithm $\mathcal{A}$. If no private user-key query on this data user id has been issued, algorithm $\mathcal{A}_0$ generates a pair of public and private user-keys (pk_{id}, sk_{id}) for this data user id, and writes them to the list L. Algorithm $\mathcal{A}_0$ issues to algorithm $\mathcal{B}_0$ a private attribute-key generation query on the set of attributes A to obtain a private attribute-key sk_A for the attribute set A, and runs ExDelegate(par_A, sk_A, A; sk_{id}) to create and return a transformation key pk_{id}^A to algorithm $\mathcal{A}$. Note that at some point, algorithm $\mathcal{A}_0$ implicitly sets the public key for a data user id^* to be pk^*, and adds $(id^*, (pk^*, \perp))$ to the list L. Algorithm $\mathcal{A}_0$ randomly chooses a transformation key $pk_{id^*}^A$, and returns it to algorithm $\mathcal{A}$. Because of the Extended Delegate property of the CP-ABE scheme **CAE** and the security of the PKE scheme **PE**, algorithm $\mathcal{A}$ cannot distinguish whether the transformation key $pk_{id^*}^A$ is randomly chosen or not.
- Challenge Phase. Algorithm $\mathcal{A}$ outputs an access structure $\mathbb{A}^*$ and two messages m_0^*, m_1^* (m_0^* and m_1^* are of

the equal length). Algorithm $\mathcal{A}_0$ sends all of them to algorithm $\mathcal{B}_0$ to generate the challenge ciphertext CT*, and returns to algorithm $\mathcal{A}$ the ciphertext CT* received from algorithm $\mathcal{B}_0$.

- Phase 2. Algorithm $\mathcal{A}$ continues issuing queries to the Private-User-Key and Transformation-Key oracles as in Phase 1, following the constraint that a query on a data user id with an attribute set A meeting the challenge access structure $\mathbb{A}^*$ should not be issued to the Transformation-Key oracle if a private user-key query on this data user id has been issued.
- Guess Phase. Algorithm $\mathcal{A}$ makes a guess b' for b. Algorithm $\mathcal{A}_0$ transmits b' to algorithm $\mathcal{B}_0$ as the output to the IND-CPA security game for the underlying CP-ABE scheme **CAE**.

Denote the event that algorithm $\mathcal{A}_0$ sets pk^* as the public user-key for a data user id^* as E. It is not hard to have the conclusion that in the view of algorithm $\mathcal{A}$, the real game and the simulation are the same except that the event E happens. Denote q_{tk} by the number of transformation key queries issued by algorithm $\mathcal{A}$. It is not difficult to conclude that the event E happens for the data user id^* with the probability $1/q_{tk}$. In this case, the transformation key $pk_{id^*}^{A^*}$ query on the data user id^* and a set of attributes A^* meeting the challenge access structure $\mathbb{A}^*$ should never be issued to algorithm $\mathcal{A}_0$. Therefore, the simulation is correct.

In summary, if algorithm $\mathcal{A}$, with a non-negligible probability ϵ, is able to win the IND-CPA security game of the PA-CPABE scheme, then algorithm $\mathcal{A}_0$, with a probability ϵ/q_{tk}, is able to win the IND-CPA security game of the underlying CP-ABE scheme **CAE**.

C. INSTANTIATION

Let $\hat{e} : G \times G \to G_1$ be a bilinear map for G being a group of a prime order p and $g \in G$ being the corresponding generator. Denote the attribute space as Z_p, and the message space as G_1. Below is the proposed concrete PA-CPABE scheme **PCAE** built on the Rouselakis-Waters CP-ABE scheme in [16] and the ElGamal PKE scheme in [18]. Note that some of the algorithms are similar to those in [13].

- Setup. The input of this algorithm is a security parameter λ. It randomly chooses a group element G with a prime order p and a generator $g \in G$, and sets a bilinear map $\hat{e} : G \times G \to G_1$. Additionally, it randomly chooses u, h, w, $v \in G$ and $\alpha \in Z_p$. Let $F_1(x) = u^x h$ be a function mapping an element $x \in Z_p$ to an element in G. It outputs $pm = (p, G, G_1, \hat{e}, g, w, v, u, h, \hat{e}(g, g)^\alpha)$ as the public parameter and $mk = \alpha$ as the master private key.
- UserKG. The input of algorithm includes the public parameter pm and a data user id. It randomly chooses $\beta_{id} \in Z_p$, and outputs $sk_{id} = \beta_{id}$ as the private user-key for the data user id. Also, it computes $pk_{id} = g^{\beta_{id}}$, and outputs pk_{id} as the public user-key for the data user id.
- PubKG. The input of this algorithm includes the public parameter pm, the master private key mk, a data user id's public user-key pk_{id} and a data user id's set of attributes $A = \{A_1, ..., A_k\}$. It randomly chooses $r, r_1, ..., r_k \in Z_p$, and computes

$$pk_1 = pk_{id}{}^{\alpha} \cdot w^r, \quad pk_2 = g^r,$$
$$pk_3^{(i)} = g^{r_i}, \quad pk_4^{(i)} = F_1(A_i)^{r_i} \cdot v^{-r}.$$

  It outputs a transformation key $pk_{id}^A = \{pk_1, pk_2, pk_3^{(i)}, pk_4^{(i)}\}_{i\in[1,k]}$ for the data user id eligible for a set of attributes A.

  Extended Delegate. Let $sk_A = (sk_1, sk_2, \{sk_3^{(i)}, sk_4^{(i)}\}_{i\in[1,k]}) = (g^\alpha \cdot w^r, g^r, \{g^{r_i}, F(A_i)^{r_i} \cdot v^{-r}\}_{i\in[1,k]})$ be the private attribute-key for the Rouselakis-Waters CP-ABE scheme [16]. Thus, we have ExDelegate$(par_A, sk_A, A'; r') = sk'_{A'} = ((g^\alpha \cdot w^r)^{r'}, (g^r)^{r'}, \{(g^{r_i})^{r'}, (F(A_i)^{r_i} \cdot v^{-r})^{r'}\}_{i\in[1,k']}) = (g^{\alpha\cdot r'} \cdot w^{r\cdot r'}, g^{r\cdot r'}, \{g^{r_i\cdot r'}, F(A_i)^{r_i\cdot r'} \cdot v^{-r\cdot r'}\}_{i\in[1,k']}) =$ ABE.KeyGen$(par_A, \alpha\cdot r', A'; (r\cdot r', \{r'\cdot r_i\}_{i\in[1,k']}))$. Since the key $sk'_{A'}$ can be written as $(g^{\alpha\cdot r'} \cdot w^r, g^r, \{g^{r_i}, F(A_i)^{r_i} \cdot v^{-r}\}_{i\in[1,k']})$ by setting $r = r \cdot r'$ and $r_i = r_i \cdot r'$, we have that $(g^{r'}, (g^{\alpha\cdot r'} \cdot w^r, g^r, \{g^{r_i}, F(A_i)^{r_i} \cdot v^{-r}\}_{i\in[1,k']})$ and $(g^{r'}, (g^{\alpha\cdot r^*} \cdot w^r, g^r, \{g^{r_i}, F(A_i)^{r_i} \cdot v^{-r}\}_{i\in[1,k']})$ are computationally indistinguishable.
- Encrypt. The input of this algorithm consists of the public parameter pm, an access structure $(\mathbb{M}, \rho)$ (assume that $\mathbb{M}$ is an $l \times n$ matrix) and a message M. It randomly chooses a vector $\overrightarrow{v} = (\mu, y_2, ..., y_n)^\perp \in Z_p^n$, of which the values are about to be used to share the secret μ. It computes $v_i = \mathbb{M}_i \cdot \overrightarrow{v}$ $(i \in [1, l])$ (denote $\mathbb{M}_i$ as the i-th row of the matrix $\mathbb{M}$). In addition, it randomly chooses $\mu_1, ..., \mu_l \in Z_p$, and computes

$$C_0 = \hat{e}(g, g)^{\alpha\mu} \cdot M, \quad C_1 = g^\mu, \quad C_2^{(i)} = w^{v_i} \cdot v^{\mu_i},$$
$$C_3^{(i)} = F_1(A_i)^{-\mu_i}, \quad C_4^{(i)} = g^{\mu_i}.$$

  It outputs CT $= ((\mathbb{M}, \rho), C_0, C_1, \{C_2^{(i)}, C_3^{(i)}, C_4^{(i)}\}_{i\in[1,l]})$ as the ciphertext.
- Transform. The input of this algorithm contains the public parameter pm, a ciphertext CT associated with an access structure $(\mathbb{M}, \rho)$ and a transformation key pk_{id}^A over attributes A of a data user id. Assume that the access structure $(\mathbb{M}, \rho)$ is satisfied by the attribute set A, and I is a set as $\{i : \rho(i) \in A\}$. Denote $\{w_i \in Z_p\}_{i\in I}$ as a class of constants which satisfies $\sum_{i\in I} w_i v_i = \mu$ when $\{v_i\}$ are valid shares of the secret μ in terms of the matrix $\mathbb{M}$. It parses CT, and calculates

$$C'_0 = \frac{\prod_{i\in I}\left(\hat{e}(C_2^{(i)}, pk_2)\hat{e}(C_3^{(i)}, pk_3^{(i)})\hat{e}(C_4^{(i)}, pk_4^{(i)})\right)^{w_i}}{\hat{e}(C_1, pk_1)}$$
$$= \frac{1}{\hat{e}(g, pk_{id}{}^{\alpha})^{\mu}}.$$

  It outputs CT$' = (C'_0, C_0)$ as the transformed ciphertext.
- Decrypt. The input of this algorithm is composed of the public parameter pm, a transformed ciphertext CT$'$ and a data user id's private user-key sk_{id}. It computes $M = (C'_0)^{1/\beta_{id}} \cdot C_0$, and outputs the message M.

Theorem 2: The given concrete PA-CPABE scheme **PCAE** is selectively IND-CPA secure under the assumption

TABLE 1. Comparison between PA-CPABE and existing solutions on reducing computation overheads in decryption of CP-ABE.

	Construction	Security	Group	Secure Channel	User Decryption
GHW [4]	Concrete	Selective	Prime-Order	Yes	E
LDGW [5]	Concrete	Selective	Prime-Order	Yes	E
LHLCX [6]	Concrete	Selective	Prime-Order	Yes	E
QDLM [9]	Concrete	Selective	Prime-Order	Yes	E
MLMCW [10]	Concrete	Selective	Prime-Order	Yes	E
FWWLY [11]	Concrete	Selective	Prime-Order	Yes	E
PA-CPABE	Generic	Selective Full	Prime-Order Composite-Order	No	E

that the Rouselakis-Waters CP-ABE scheme is selectively IND-CPA secure and satisfies the Extended Delegate property, and the ElGamal PKE scheme is IND-CPA secure.

Proof: It has been proved in [16] that the Rouselakis-Waters CP-ABE scheme is selectively IND-CPA secure, and the ElGamal PKE scheme is known to be IND-CPA secure. In addition, the Rouselakis-Waters CP-ABE scheme satisfies the Extended Delegate property. Therefore, the given concrete PA-CPABE scheme **PCAE** is selectively IND-CPA secure on the basis of Theorem 1.

D. DISCUSSIONS

Our proposed generic construction for PA-CPABE can be improved as follows.

- Verifiable and Outsourced Decryption. The proxy (i.e., an edge device) cannot be trusted in a PA-CPABE scheme and may falsely execute the calculation, so it is crucial to check whether the transformation has been correctly conducted. Though the transformation key stored by the proxy is public, and any entity can verify the correctness of the transformation, it is still useful to have ABE with verifiable and outsourced decryption (ABE-VOD) schemes such that the data user is empowered with the capability to efficiently verify whether the transformation he/she has received from the proxy is correct or not as in some cases incorrect calculation might cause disastrous outcomes. The generic construction on ABE-VOD in either [9] or [10], built from ABE schemes supporting outsourced decryption, can be applied to the proposed generic PA-CPABE construction to achieve efficient verification of the transformation executed by the proxy when the data user attempts to get the plaintext by running the decryption algorithm on the transformed ciphertext.
- Attribute and User Revocation. An edge computing network always involves a great number of data users whose statuses might not be immutable and could regularly change, because some data users may leave the edge computing network after a certain time period, yet the attributes of data users may differ over time. It is beneficial for a PA-CPABE scheme to be equipped with an efficient revocation mechanism such that data users as well as the attributes possessed by data users in the edge computing network can be selectively revoked by the trusted authority (e.g., the KGC in PA-CPABE). There exist generic techniques to achieve the revocation of data users' attributes and data users in ABE schemes (e.g., [20], [21]) under the setting of an untrusted (or semi-trusted) third party (e.g., [13], [21]), which can be applied to the proposed PA-CPABE construction to efficiently revoke attributes of data users and data users in an edge computing network.

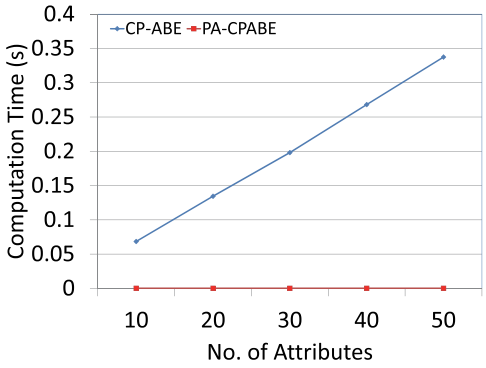
V. PERFORMANCE EVALUATION

After doing a comparison between several existing attribute-based encryption with outsourced decryption (ABE-OD) schemes and the proposed notion of proxy-aided ciphertext-policy attribute-based encryption (PA-CPABE), we implement the given instantiation for PA-CPABE to evaluate its performance in this section.

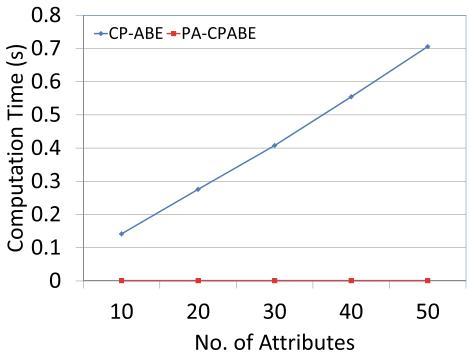
A. COMPARISON

Recall that there have been plenty of constructions (e.g., [4]–[6], [9]–[11]) that target to mitigate the data user's computation workload in the decryption phase of attribute-based encryption (ABE) schemes by outsourcing the decryption cost to a proxy such that only one exponentiation operation needs to be performed by a privileged data user to decrypt a ciphertext. The first ABE with outsourced decryption (ABE-OD) scheme was put forward by Green *et al.* [4], where the proxy partially decrypts (i.e., transforms) the ciphertext and transmits to the data user the transformed ciphertext for decryption. The notion of ABE with verifiable and outsourced decryption (ABE-VOD) was brought in by Lai *et al.* [5] and Li *et al.* [6], respectively, providing efficient verification on the accurateness of the transformation executed by the proxy. Qin *et al.* [9] and Mao *et al.* [10] presented generic approaches of transforming ABE-OD schemes to ABE-VOD schemes, respectively. Fan *et al.* [11] added the revocation function to a concrete ABE-VOD scheme and applied the resulting scheme to a fog-cloud (i.e., edge) computing network.

Table 1 compares the proposed PA-CPABE scheme and several existing works related to outsourcing the workloads resulted from decrypting ABE ciphertexts to a third party, where "E" denotes exponentiation. It is straightforward to see that there are no secure channels required in the proposed PA-CPABE construction for the delivery of private (or decryption) keys from the KGC to each data user in the edge computing network, while all other existing constructions



(a)



(b)

FIGURE 3. Average computation time of the data user in decrypting a ciphertext. (a) SS512. (a) MNT159.

need secure channels to distribute private keys to data users to achieve security. In addition, the existing solutions on ABE with outsourced decryption are concrete schemes, while the proposed construction on PA-CPABE is generic which can convert any CP-ABE scheme satisfying certain properties to a PA-CPABE scheme.

B. EXPERIMENTAL RESULTS

The given instantiation of PA-CPABE and its underlying CP-ABE scheme in [16] are implemented in a framework called Charm [22]. In addition to the Charm framework, the Python package and the PBC library are installed for certain cryptographic operations. All experiments are executed on an all-in-one desktop with the 8GB RAM and the Intel Core i5-6500 CPU @ 3.2GHz running the 64-bit Ubuntu 16.04 over a VMware Workstation Player which is set with the 1GB RAM [17].

To provide an eighty-bit security level, the simulation is conducted under two elliptic curves knowns as SS512 and MNT159.[3] In the underlying CP-ABE scheme of the given concrete
PA-CPABE scheme and the given concrete PA-CPABE scheme, the average computation time spent by a data user on the decryption of ciphertexts ascribed to access structures consisting of ten to fifty attributes is summarized as in Fig. 3 (note that the average computational cost of the proxy in decrypting a ciphertext in the given concrete PA-CPABE scheme is similar to that of the data user in decrypting a ciphertext in the underlying CP-ABE scheme [16]). Concerning the underlying CP-ABE scheme [16], the average computation time spent by a data user on running the decryption on ciphertexts over access structures containing ten to fifty attributes and private attribute-keys with ten to fifty attributes ranges from 0.07s to 0.34s with respect to the SS512 curve and 0.15s to 0.71s with respect to the MNT159 curve, respectively. For the given concrete PA-CPABE scheme, the average computation time spent by a data user (using a private user-key) with ten to fifty attributes on the decryption of ciphertexts associated with access structures including ten to fifty attributes is about 0.2ms regarding the SS512 curve and 1.0ms regarding the MNT159 curve, respectively. It is clear to find from Fig. 3 that PA-CPABE has the capability of significantly reducing the computational overheads of data users in decrypting ciphertexts, where the computational cost of a data user in doing the decryption on ciphertexts is independent to the size of attributes related to the ciphertexts and the attribute-keys.

[3]Note that MNT159 is known as an asymmetric Type 3 pairing, while SS512 is known as a symmetric Type 1 pairing.

VI. CONCLUSIONS

The Internet of Things (IoT) devices constantly generate data, and require the data analysis to be rapid, which cannot be provided by the traditional cloud computing architecture. With the target of analyzing the IoT data close to the devices that generate and operate on the data, edge computing has been introduced for the extension to the edge of the network from cloud computing. Though edge computing facilitates cloud computing in addressing the latency problem of data processing, it also brings more security and privacy issues to the existing cloud computing network. Due to the fact that attribute-based encryption (ABE) supports fine-grained (or scalable) access control for data items in encrypted forms, ABE has been widely believed to be an ideal solution to protect data security and privacy for scenarios of cloud computing. To achieve fine-grained access control for the edge computing environment, in this paper, we proposed a notion named proxy-aided ciphertext-policy attribute-based encryption (PA-CPABE). After describing a generic construction of PA-CPABE, we formally analyzed its security. In addition, we presented and implemented an instantiation of PA-CPABE to evaluate its efficiency.

REFERENCES

[1] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," *IEEE Internet Things J.*, vol. 3, no. 5, pp. 637–646, Oct. 2016.

[2] F. Bonomi, R. A. Milito, J. Zhu, and S. Addepalli, "Fog computing and its role in the Internet of Things," in *Proc. 1st Ed. MCC Workshop Mobile Cloud Comput.*, Helsinki, Finland, Aug. 2012, pp. 13–16.

[3] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Advances Cryptology—EUROCRYPT* (Lecture Notes in Computer Science). Aarhus, Denmark: Springer, May 2005, pp. 457–473.

[4] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of ABE ciphertexts," in *Proc. 20th USENIX Secur. Symp.* San Francisco, CA, USA: USENIX Assoc., Aug. 2011, pp. 1–16.
[5] J. Lai, R. H. Deng, C. Guan, and J. Weng, "Attribute-based encryption with verifiable outsourced decryption," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 8, pp. 1343–1354, Aug. 2013.
[6] J. Li, X. Huang, J. Li, X. Chen, and Y. Xiang, "Securely outsourcing attribute-based encryption with checkability," *IEEE Trans. Parallel Distrib. Syst.*, vol. 25, no. 8, pp. 2201–2210, Aug. 2014.
[7] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Proc. IEEE Symp. Secur. Privacy (SP)*, Berkeley, CA, USA, May 2007, pp. 321–334.
[8] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Public Key Cryptography—PKC* (Lecture Notes in Computer Science). Taormina, Italy: Springer, 2011, pp. 53–70.
[9] B. Qin, R. H. Deng, S. Liu, and S. Ma, "Attribute-based encryption with efficient verifiable outsourced decryption," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 7, pp. 1384–1393, Jul. 2015.
[10] X. Mao, J. Lai, Q. Mei, K. Chen, and J. Weng, "Generic and efficient constructions of attribute-based encryption with verifiable outsourced decryption," *IEEE Trans. Depend. Sec. Comput.*, vol. 13, no. 5, pp. 533–546, May 2016.
[11] K. Fan, J. Wang, X. Wang, H. Li, and Y. Yang, "A secure and verifiable outsourced access control scheme in fog-cloud computing," *Sensors*, vol. 17, no. 7, p. 1695, 2017.
[12] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Advances Cryptology—EUROCRYPT* (Lecture Notes in Computer Science). Tallinn, Estonia: Springer, 2011, pp. 568–588.
[13] H. Cui, R. H. Deng, Y. Li, and B. Qin, "Server-aided revocable attribute-based encryption," in *Computer Security—ESORICS* (Lecture Notes in Computer Science). Heraklion, Greece: Springer, 2016, pp. 570–587.
[14] H. Cui, R. H. Deng, Y. Li, and G. Wu, "Attribute-based storage supporting secure deduplication of encrypted data in cloud," *IEEE Trans. Big Data*, to be published, doi: 10.1109/TBDATA.2017.2656120.
[15] A. Beimel, "Secure schemes for secret sharing and key distribution," Ph.D. dissertation, Israel Inst. Technol., Haifa, Israel, Jun. 1996.
[16] Y. Rouselakis and B. Waters, "Practical constructions and new proof methods for large universe attribute-based encryption," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur. (CCS)*, Berlin, Germany, Nov. 2013, pp. 463–474.
[17] H. Cui, R. H. Deng, J. Lai, X. Yi, and S. Nepal, "An efficient and expressive ciphertext-policy attribute-based encryption scheme with partially hidden access structures, revisited," *Comput. Netw.*, vol. 133, pp. 157–165, Mar. 2018.
[18] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms," *IEEE Trans. Inf. Theory*, vol. IT-31, no. 4, pp. 469–472, Jul. 1985.
[19] H. Cui, M. H. Au, B. Qin, R. H. Deng, and X. Yi, "Fuzzy public-key encryption based on biometric data," in *Provable Security* (Lecture Notes in Computer Science). Xi'an, China: Springer, Oct. 2017, pp. 400–409.
[20] A. Sahai, H. Seyalioglu, and B. Waters, "Dynamic credentials and ciphertext delegation for attribute-based encryption," in *Advances in Cryptology—CRYPTO* (Lecture Notes in Computer Science). Santa Barbara, CA, USA: Springer, 2012, pp. 199–217.
[21] H. Cui, R. H. Deng, X. Ding, and Y. Li, "Attribute-based encryption with granular revocation," in *Security and Privacy in Communication Networks* (Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering), vol. 198. Guangzhou, China: Springer, Oct. 2016, pp. 165–181.
[22] J. A. Akinyele *et al.*, "Charm: A framework for rapidly prototyping cryptosystems," *J. Cryptograph. Eng.*, vol. 3, no. 2, pp. 111–128, 2013.

HUI CUI received the Ph.D. degree from the School of Computing and Information Technology, University of Wollongong, Australia. She is currently a Research Fellow with the School of Science, Royal Melbourne Institute of Technology University, Australia.

XUN YI is currently a Professor with the School of Science, RMIT University, Australia. He has authored over 150 research papers in international conferences and journals such as the IEEE Transactions on Knowledge and Data Engineering, the IEEE Transactions on Wireless Communications, the IEEE Transactions on Dependable and Secure Computing, and the IEEE Transactions on Circuits and Systems. His research interests include data privacy, cloud security, cybersecurity, wireless and mobile security, and applied cryptography. He has been an Associate Editor of the IEEE Transactions on Dependable and Secure Computing since 2014.

SURYA NEPAL received the bachelor's degree from the National Institute of Technology, Surat, India, the master's degree from the Asian Institute of Technology, Bangkok, Thailand, and the Ph.D. degree from RMIT University, Australia. He joined CSIRO in 2000. He is currently a Principal Research Scientist with Data61, CSIRO. His research at CSIRO includes multimedia databases, Web services and service-oriented architectures, social networks and security, privacy and trust in collaborative environment and cloud systems, Internet of Things (IoT), and big data platforms. He is the leader of the Distributed System Security Group, Data61, CSIRO, comprising 10 research staff and a number of Ph.D. students. He has published over 200 publications, in which many of them are published in the top international journals and conferences such as VLDB, ICDE, ICWS, SCC, CoopIS, ICSOC, the IEEE Transactions on Services Computing, the IEEE Transactions on Parallel and Distributed Processing, the *ACM Transactions on Internet Technology*, and the *ACM Computing Surveys*. His main research interest is in the development and implementation of technologies in the area of distributed systems including Web services, cloud computing, IoT, and big data, with a specific focus on security, privacy, and trust. He is currently one of the associate editors of the IEEE Transactions on Services Computing.