Privacy-preserving multi-user encrypted access control scheme for cloud-assisted IOT applications

Maryline Laurent, Nesrine Kaaniche

Privacy-preserving Multi-user Encrypted Access Control Scheme for Cloud-assisted IoT applications

Nesrine Kaaniche
SAMOVAR, Telecom SudParis,
University Paris-Saclay, France

Maryline Laurent
SAMOVAR, Telecom SudParis,
University Paris-Saclay, France

***Abstract*—In this paper, we present a privacy preserving encrypted access control scheme to aggregate data for Cloud assisted IoT applications. Our scheme is based on attribute based encryption mechanisms and consists in enciphering a set of data contents, with respect to sub-sets of a general access policy. As such, the gateway is able to decrypt the resulting aggregated data only if it holds the matching certified attributes and it has received a sufficient number of partial ciphertexts. Our construction has several advantages. First, it provides a fine-grained access to aggregated data contents that are enciphered by different multiple encrypting entities. Second, it provides a privacy preserving encryption process, such that a curious gateway can neither identify the enciphering IoT device nor decipher single data chunks. Third, our concrete construction provides low computation and communication costs, adapted to resource-constrained devices, compared to most closely related schemes.**

***Keywords*-Cloud-assisted IoT, Attribute based Encryption, access control, aggregation, privacy.**

I. Introduction

During the last decade, Internet of Things (IoT) approached our lives in many ways, thanks to the availability of several communication systems that have been increasingly deployed, mainly for smart monitoring and control applications. According to the U.S. International Data Corporation (IDC) [16], IoT is experiencing a fast expansion with an estimated 16.7% growth year over year in 2017 to reach nearly $1.4 trillion by 2021. Typically, IoT deployments permit a collection of smart objects, generally considered as resource-constrained devices, to communicate among themselves and also to an external gateway or a base-station, for further processing. Thus, cloud computing-enhanced IoT has recently received considerable attention, as the gateways deployed at the network edge can not only provide low latency, location awareness but also improve real-time and quality of services in IoT application scenarios, while protecting the secrecy of transmitted sensed data and preserving the privacy of devices.

Privacy-preserving data aggregation is one of the typical cloud assisted IoT applications, and many privacy-preserving data aggregation schemes have been proposed in the past years for ensure reliable processing of data collected from different IoT devices. Indeed, in IoT applications, data are always transmitted, stored and dynamically shared through heterogeneous and distributed networks [10], [7]. Consequently, encryption and access control mechanisms are important in order to prevent unauthorized entities from accessing data [13], [5], [2].

Contributions — In this paper, we introduce a new privacy preserving multi-user encrypted access control scheme to aggregated data, based on the use of Ciphertext-Policy Attribute based Encryption (CP-ABE) mechanisms [3], that ensures privacy-preserving processing for IoT applications. CP-ABE is a powerful primitive that provides an encrypted access control mechanism, based on fine-grained access structures. However, this encryption technique suffers from heavy computation overhead which is burdensome for resource-constrained devices. As such, our construction consists in performing in a secure collaborative manner the encryption of a set of data chunks. The main idea behind our encrypted access control scheme relies on the distribution of the enciphering operation among different devices, with respect to sub-sets of a general access structure. That is, each device encrypts its input data and sends the partial enciphered information to an untrusted gateway. Thus, if this latter holds the matching certified attributes and receives a sufficient number of partial ciphertexts then it will be able to aggregate and then decrypt the computation result.

Paper organization — Section II introduces a motivating use-case and highlights the design goals. Section III presents Attribute based Encryption mechanisms and reviews related work. Section IV details the network and security models. Section V provides useful theorems and presents our concrete construction. Section VI gives a security analysis and section VII discusses the performances of the proposed scheme. SectionVIII gives possible improvements before concluding in section IX.

II. Problem Statement

In this section, we first give a motivating scenario (cf. section II-A). Then, we highlight the security and functional design goals (cf. section II-B).

A. Motivating Scenario

The use of IoT devices is gaining an expanding interest mainly for remote health monitoring which is widely applied by several health organizations to provide better personalized services. Indeed, Wireless Body Area Networks (WBANs)

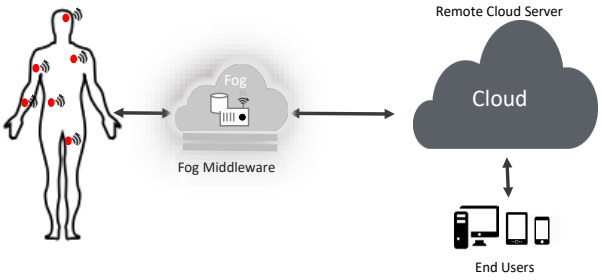

Fig. 1. WBAN Fog based Architecture

consist of intelligent and resource-constrained devices that can be placed on the patient body to monitor his health status. WBANs devices monitor various parameters such as blood gas, blood pressure, pulse rate, temperature, electrocardiogram (ECG), and electroencephalogram (EEG) (cf. Figure 1).

While dealing with medical data, security and privacy are among the most challenging issues. For instance, the EGC can reflect users some specific behaviors, such as sleeping, having meals and so on. In the sequel, the reveal of such health data might violate users privacy. Therefore, how to efficiently aggregate different types of data and preserve users privacy is still challenging in cloud assisted WBANs. Hence, the data transmitted in health-care applications should be authenticated and secured against malicious access. In addition, developing security techniques to secure patients' data should fulfill low communication and computation cost requirements. Thus, the resource consumption in a WBAN is tied up to the amount of data being processed, stored, and transmitted. Hence, aggregation has been often presented as a solution to take advantage of the aggregating node computing capabilities in order to process sensed data. In the architecture depicted in Figure 1, the WBAN gateways are responsible for performing aggregation of the data received from the smart things deployed on the body of the patient, then to transmit deciphered data to the cloud. This latter receives all IoT devices' aggregated data via the fog device, and makes some data analytics according to some application requirements. Thus, aggregated data should be accessed by a set of authorized gateways, thus ensuring a flexible access control mechanism.

B. Design Goals

The design of our encrypted access control scheme is motivated by providing the support of both robustness and efficiency while fulfilling the following properties:

- ***multi-user fine grained access control –*** our proposal should ensure flexible access policies among different groups of users. In addition, the aggregated accessed data may be enciphered with different IoT devices.
- ***data confidentiality –*** the proposed scheme has to protect the secrecy of encrypted data contents against malicious entities, even in case of collusions.
- ***privacy –*** preserving users' privacy is multifold. First, it is useful in a context where anonymity should be enforced to forbid any user's identification or personal information leakages (e.g. sex, age, address). Second, unauthorized users should not be able to obtain any information about honest users' inputs, or to link the obfuscated content to a specific entity. Third, access to obfuscated data should not reveal identifying information of the requesting entity.
- ***low processing and communication costs –*** the designed algorithm should have low computational complexity and reduced communication overhead, to efficiently adapt to IoT applications.

III. Attribute Based Encryption

In 2005, Sahai and Waters introduced the concept of Attribute Based Encryption [14], as a generalization of Identity based Encryption (IBE), introduced by Shamir [15], in 1984, with the original idea to provide public and private key pairs based on user's unique identity. ABE appears as a promising technique, designed for ensuring encrypted fine grained access control, for many users based on general access policies. In ABE, both users' private keys and ciphertexts are associated with a set of attributes or a structure over attributes and the user is able to decrypt a ciphertext if there is a match between his private key and the ciphertext [3]. For instance, Goyal et al. [6] distinguish two ABE categories, namely: Key-Policy ABE (KP-ABE) and Ciphertext-Policy ABE (CP-ABE). There exist some differences between KP-ABE and CP-ABE mechanisms. In fact, for KP-ABE schemes, the secret key of user U is derived with respect to an access structure Γ_u, and the ciphertext is generated according to a set of attributes $\mathcal{S}_{CT}$. As such, the user U can decrypt the ciphertext CT if the set of attributes $\mathcal{S}_{CT}$ satisfy the access tree associated with the user' secret keys. On the contrary, for CP-ABE schemes, the secret key of user U is associated with a set of attributes $\mathcal{S}_u$, and the ciphertext is created with respect to an access tree Γ. As such, U can decrypt the ciphertext CT if the set of attributes $\mathcal{S}_u$ associated to his secret key satisfy the access tree.

CP-ABE is a very interesting primitive, that permits to ensure fine-grained access control, by directly embedding the access policy on the ciphertext. However, its processing and communication overheads are still highly resource-consuming, mainly for IoT devices [9]. In [18], Touati et al. proposed a cooperative CP-ABE scheme, such that resource-constrained devices can delegate the most consuming operations to unconstrained nodes. However, their scheme does still require a central trusted party that has to derive the secret enciphering key. As such, [18] cannot be adapted for fully distributed environments. Afterwards, the authors presented a collaborative KP-ABE scheme for Cloud assisted IoT applications [17]. Their construction is based on the use of computing and storage capacities of cloud servers and assistant nodes to perform heavy operations and save resources of tiny smart objects.

In the same vein, Oualha and Nguyen extended the CP-ABE construction proposed by Bethencourt et al. in [3], to apply the scheme in the context of resource-constrained IoT devices [12]. The authors proposed a hybrid approach alternating pre-computations and on-demand computations can be devised

to overcome the issue of limited storage and processing capacities' of IoT. Recently, in 2017, Odelu et al. presented a new RSA-based CP-ABE scheme with constant size ciphertexts (CSKC) and low complexity for both decryption and encryption processes [11]. However, these proposed schemes do not consider enciphering a set of different data chunks, hence providing a multi-encryptor setting and enabling the deciphering entity to only decrypt the aggregated data content.

IV. Design Models

In this section, we formalize our system (cf. section IV-A) and security (cf. section IV-B) models.

A. System Model

For our system model, five entities are identified: an attribute authority (AA), a manager (GM), a set of IoT devices $\{U_i\}_{\{i=1,\cdots,n\}}$, a set of sink aggregating entities $\{S_j\}_{\{j=1,\cdots,m\}}$ and a selected IoT trusted node defined such as:

- **attribute authority** AA – is responsible for bootstrapping the whole system in the initialization phase. We assume that AA is a trusted entity. It also issues certified attributes and related secret keys for the aggregating entities and IoT devices.
- **manager** GM – is in charge of generating the general access predicate used to encrypt data contents as well as the aggregating function for each specific cloud-IoT application.
- **aggregating entity** (S_j) – is considered as a local network gateway. It is responsible for collecting and deciphering aggregated data contents.
- **IoT device** (U_i) – collects and encrypts sensed data before forwarding them to the aggregating entity.
- **trusted IoT node** (U_s) – is a trusted selected IoT device and it periodically assigns to each involved IoT device U_i a sub-access predicate to be used for encrypting data.

Our scheme is composed of six randomized algorithms based on three phases, namely SysInit, Co-Encrypt and Aggregation. During the SysInit phase, the central attribute authority generates public system parameters and derives secret keys of aggregating entities, based on the `setup` and `keygen` algorithms, respectively. This phase is executed once for the system initialization.
The Co-Encrypt phase starts when the encryption of data chunks are requested. The second phase relies on two different algorithms, namely `accshare`, executed by the trusted IoT node and `encpart`, performed by each participating entity.
The Aggregation phase permits an aggregating authorized entity to build the resulting ciphertext and to decrypt it with respect to its associated attributes. This phase is based on two different algorithms, i.e; `aggregate` and `decrypt`. The different algorithms are detailed in Section V.

B. Security Model

For designing a secure and privacy preserving encrypted access control scheme, we consider two main adversaries.
We first point out the case of *semi-honest* gateway. That is, a semi-honest adversary provides proper inputs or outputs, at each step of the protocol, and properly performs any expected calculations, but it may attempt to gain extra information from the protocol. As such, we consider the *semi-honest* threat model against the privacy requirement with respect to the anonymity and unlinkability properties.
Second, we consider the case of *malicious* adversaries, trying to override their rights. That is, malicious users may attempt to deviate from the protocol or to provide invalid inputs. For instance, a malicious party can also influence other parties to deviate from the protocol by substituting their local inputs in order to get information about honest parties' inputs. Likewise, the malicious party may refuse to participate or leave the protocol before the end. As such, we consider the malicious user security model mainly against the confidentiality property.

V. Concrete construction of our Privacy preserving Access Control scheme to Aggregated Encrypted Data

Our privacy-preserving access control scheme is relying on ABE in the sense that clients' keys and decryption capabilities are related to the attributes they possess. In our proposal, the ciphertext is generated with respect to a set of messages and the aggregating gateway's credentials (certified attributes) determine his capability to decrypt the result of a computed function f over the enciphered set of messages. That is, the manager GM defines the aggregating function f and the general access policy, w.r.t. each application requirements.
Our scheme consists in performing the encryption of a set of different data chunks collected by a set of IoT devices. It relies on the distribution of the enciphering operation among different devices, with respect to sub-sets of a general access structure Γ, such that $\Gamma = \cup_{i\in[1,n]}\gamma_i$. More precisely, each device U_i encrypts its sensed data message m_i relying on a sub access structure γ_i and sends the partial enciphered information to an untrusted aggregating gateway S_j.

In the following, we introduce preliminaries before detailing our concrete construction.

A. Mathematical Background

In this section, we provide some prerequisites, namely access structures and bilinear maps.

Definition 5.1: (**Access Structure** [1]) Let $\mathcal{P} = \{P_1, P_2, \cdots, P_n\}$ be a set of parties, and a collection $\mathbb{A} \subseteq 2^{\{P_1,P_2,\cdots,P_n\}}$ is called monotone if $\forall B, C \subseteq 2^{\{P_1,P_2,\cdots,P_n\}}$: if $B \in \mathbb{A}$ and $B \subseteq C$ then $C \in \mathbb{A}$. An access structure is a collection $\mathbb{A}$ of non-empty subsets of $\{P_1, P_2, \cdots, P_n\}$; i.e. $\mathbb{A} \subseteq 2^{\{P_1,P_2,\cdots,P_n\}} \setminus \{\emptyset\}$. The sets in $\mathbb{A}$ are called authorized sets, and the sets not in $\mathbb{A}$ are called unauthorized sets. Note that in several ABE schemes, these parties are considered as the attributes.

Definition 5.2: (**Bilinear Maps**) Let $\mathbb{G}_1$, $\mathbb{G}_2$, and $\mathbb{G}_T$ be three cyclic groups of prime order p. Let g_1, g_2 be generators

of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively . A map $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ is said to be bilinear if it satisfies the following properties: (i) bilinearity: for all $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2$, (ii) non-degeneracy: $\hat{e}(g_1, g_2) \neq 1$ and (iii) there is an efficient algorithm to compute $\hat{e}(g_1, g_2)$ for any $g_1 \in \mathbb{G}_1$ and $g_2 \in \mathbb{G}_1$.

B. Access Tree Model

Let Γ be a tree representing the access structure, following the definition given in [3]. Indeed, each non-leaf node of Γ is described by the number of its children num_x and a threshold value t_x, where $1 \leq t_x \leq num_x$. If the threshold value $t_x = num_x$, then it is an "AND "gate, otherwise it is an "OR "gate.

As introduced in [3], three additional functions are defined namely $\texttt{parent}(x)$, $\texttt{att}(x)$ and $\texttt{index}(x)$. The $\texttt{parent}(x)$ function denotes the parent of the node x, the $\texttt{att}(x)$ denotes the attributes associated with the leaf node x and the $\texttt{index}(x)$ denotes a number associated with the node.

Satisfying an access tree – Let Γ be an access tree rooted at the node r. We denote by Γ_x the subtree of Γ rooted at the node x, such that $\Gamma_r = \Gamma$. As such, $\Gamma_x(\mathcal{S}) = 1$ if a set of attributes denoted $\mathcal{S}$ satisfies the access tree Γ_x. If x is a non-leaf node, $\Gamma_x(\mathcal{S})$ is computed by recursively evaluating the subtrees $\Gamma_{x'}(\mathcal{S})$ for all its x' children. $\Gamma_x(\mathcal{S})$ returns 1 if at least t_x children returns 1. If x is a leaf node, then $\Gamma_x(\mathcal{S}) = 1$ if $\texttt{att}(x) \in \mathcal{S}$.

C. Concrete Construction

Our construction is an access control scheme for multi-user encryption settings, based on the ciphertext policy attribute based encryption algorithm introduced in [3]. To support the multi-user encryption feature, the proposed scheme considers that the encryption process is performed by each participating device U_i relying on a secret s_i with respect to a dedicated access subtree γ_i, while the decryption process should be performed with respect to a general access structure $\Gamma = \cup_{i \in [1,n]} \gamma_i$, where n is the number of sub-trees.

Let $\mathcal{U} = \{U_1, \cdots, U_n\}$ be a set of IoT devices such that $|\mathcal{U}| = n$. Each participant U_i has to encrypt a message m_i (i.e; $i \in [1, n]$), w.r.t. an access subtree γ_i and publish the partial ciphertext CT_i to the aggregating entity S_j. The encryption of each message m_i by U_i relies on a different secret share s_i. The decryption of the resulting message M, defined as $M = f(\{m_i\}_{\{i \in [1,n]\}})$ may be executed by any S_j [1].

In the following, we detail the different algorithms of our construction, thus we suppose that GM has already set up the general access tree Γ and the aggregating function f. That is, during the SysInit phase, AA executes `setup` and `keygen` defined as follows:

- `setup` – this algorithm is executed by the attribute authority. It takes as input the security parameter κ and outputs the global public parameters pp and the master secret key msk. the `setup` algorithm first defines a bilinear setting $(\hat{e}, \mathbb{G}_1, \mathbb{G}_2, p, g)$, such as $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_1 \leftarrow \mathbb{G}_2$ and g of a generator of $\mathbb{G}_1$. Then, it selects at random an integer $\alpha \in \mathbb{Z}$. Let $h := g^\beta \in \mathbb{G}_1$ and $\mathcal{H}$ be a cryptographic hash function, such that $\mathcal{H} : \mathbb{S} \rightarrow \mathbb{G}_1$. The global parameters of our system are as follows:
$$\texttt{pp} = \{\mathbb{G}_1, \mathbb{G}_2, \hat{e}, p, g, h, \hat{e}(g, g)^\alpha, \mathcal{H}\}$$
We note that the master secret key of the attribute authority is the couple $msk = \{\beta, g^\alpha\}$.
- `keygen` – this algorithm is performed by the attribute authority, as detailed in [3], in order to derive private keys associated with a set of attributes $\mathcal{S} = \{a_j\}_{\{j \in [1, |\mathcal{S}|]\}}$ of a requesting entity. That is, AA selects a set of random values $\{r, \{r_j\}_{\{j \in [1, |\mathcal{S}|]\}}\}$, related to the requesting entity and to its associated attributes $\mathcal{S} = \{a_j\}_{\{j \in [1, |\mathcal{S}|]\}}$. The private key sk is defined as follows:
$$sk = \{D = g^{(\alpha + r)/\beta},$$
$$\forall a_j \in \mathcal{S}, D_j = g^r \cdot \mathcal{H}(a_j)^{r_j}, D'_j = g^{r_j}\}$$

During the Co-Encrypt phase, two algorithms `accshare` and `encpart` are performed by the trusted device U_s and each participating entity U_i respectively:

- `accshare` – this algorithm is performed by U_s, in order to set up required enciphering elements for the n participating users $\{U_i\}_{i \in [1,n]}$. Note that this algorithm is performed periodically, for saving the energy consumption of involved IoT devices. That is, U_s defines the access subtree γ_i associated with each user U_i, such that $\{(U_i, \gamma_i)\}$, where $i \in [1, n]$ and $\Gamma = \cup_{i \in [1,n]} \gamma_i$. Then, it chooses a secret deciphering key s such as $s = \sum_{i=1}^{n} s_i$ and s_i is the secret share associated with each device U_i. Finally, U_s sets $C = h^s$ and outputs :
$$\{\{(\gamma_i, s_i)\}_{i \in [1,n]}, C\}$$
Recall that each (γ_i, s_i) is privately shared with the corresponding U_i, while C is published for further processing by the aggregating entities.
- `encpart` – this algorithm is run by each participating IoT device U_i, in order to derive a partial ciphertext CT_i for a message m_i, with respect to an access sub-tree γ_i, a secret share s_i and the aggregating function f. U_i proceeds similarly as explained in [3]. That is, for each node x of its associated sub-tree γ_i, U_i defines a polynomial q_x of degree $d_x = k_x - 1$ where k_x is the threshold value of the node x and $q_x(0) = q_{\texttt{parent}(x)}(\texttt{index}(x))$.
Let $\mathcal{L} = \{Y_1, Y_2, \cdots, Y_n\}$ be the set of leaf nodes of the global access tree Γ, where each subset Y_i (i.e; $i \in [1, n]$) is assigned to a user U_i, with respect to its associated subtree γ_i. Let q_{y_j} be the polynomial assigned to each leaf node $y_j \in Y_i$ (i.e; $j \in [1, |Y_i|]$), associated to the IoT device U_i. For each leaf node $y_j \in Y_i$ (i.e; $j \in [1, |Y_i|]$), the IoT U_i computes two group elements $C_{i,j}$ and $C'_{i,j}$, such that $C_{i,j} = g^{q_{y_j}(0)}$ and $C'_{i,j} = \mathcal{H}(\texttt{att}(y))^{q_{y_j}(0)}$.
Afterwards, U_i encrypts the message m_i with respect to γ_i and outputs the partial ciphertext CT_i such that:
$$CT_i = \{\tilde{C}_i, \forall y_j \in Y_i, j \in [1, |Y_i|] : C_{i,j}, C'_{i,j}\}$$

[1]For ease of presentation, f refers to the multiplication function. Yet, in general use cases, f may be any bijective and homomorphic function.

where, $\tilde{C}_i$ depends on the aggregating function f pointed out by the manager. In the sequel, the encrypted message CT_i is defined as follows:

$$\begin{cases} \tilde{C}_i = m_i \cdot \hat{e}(g,g)^{\alpha s_i} \\ C_{i,j} = g^{q_{y_j}(0)} \\ C'_{i,j} = \mathcal{H}(\texttt{att}(y))^{q_{y_j}(0)} \end{cases}$$

Recall that the partial ciphertext CT_i associated to each IoT device U_i does not include γ_i.

Remark 5.3: Processing cost efficiency for IoT devices — Considering resource constraints of IoT devices, in terms of storage and processing, our scheme assumes that several elementary functions may be used for several encryption sessions (i.e., computation of polynomials and encryption of leaves' nodes). As such, only a few number of exponentiations and multiplication is required by each single IoT device.
In addition, for devices with a very limited lifetime and low storage and computation capacities, it is assumed that these objects are pre-configured by the manufacturer with a secret s_i. As such, once installed in the system, this secret is shared with the IoT trusted node. As a consequence, the IoT trusted device computes the ciphertext element C with respect to the secrets of involved IoT devices, while selecting different combinations of devices for each session. And, the communication overheads are highly reduced during the whole lifetime of resource-constained IoT devices, while avoiding exchanging secrets and re-computing some ciphertexts' elements periodically .

During the AGGREGATE phase, two algorithms `aggregate` and `decrypt` are performed by the aggregating node S_j:

- `aggregate` – this algorithm is executed by the aggregating entity S_j, upon receiving the n ciphertexts from the participating entities $\{U_i\}$, where $i \in [1,n]$. It permits to build a global ciphertext, with respect to the global public access tree Γ and the aggregating function f. The `aggregate` algorithm outputs an aggregated global ciphertext CT, defined as:

$$CT = \{\tilde{C}, C, \forall Y_i \in \mathcal{L}, \forall y_j \in Y_i : C_{i,j}, C'_{i,j}\}$$

where C is received from the trusted IoT node and $\tilde{C}$ is computed as follows:

$$\prod_{i\in[1,n]} \tilde{C}_i = \prod_{i=1}^{n} m_i \cdot \hat{e}(g,g)^{\alpha \sum_{i=1}^{n} s_i} = M.\hat{e}(g,g)^{\alpha s}$$

In the following, we denote by C_y and C'_y each element $C_{i,j}$ and $C'_{i,j}$, respectively.
- `decrypt` – this algorithm, executed by the aggregating entity S_j, permits to decrypt the aggregated ciphertext CT, including the global access structure Γ and the private key sk associated with $\mathcal{S}$. The `decrypt` algorithm is mainly based on the Bethencourt et al. construction [3]. To do so, for decrypting the aggregated ciphertext with respect to the function f, the aggregating entity uses a recursive algorithm $\texttt{DecryptNode}(CT, sk, x)$, as defined in [3] which takes as input the global ciphertext CT, the private key sk, associated with a set of attributes $\mathcal{S}$, and a node x from the global access tree Γ.
If x is a leaf node, then we denote by $a_i = \texttt{att}(x)$.
If the attribute $a_i \in \mathcal{S}$, then

$$\texttt{DecryptNode}(CT, sk, x) = \frac{\hat{e}(D_i, C_x)}{\hat{e}(D'_i, C'_x)} = \hat{e}(g,g)^{rq_x(0)}$$

else, $\texttt{DecryptNode}(CT, sk, x) = \perp$. If x is a non-leaf node, then $\texttt{DecryptNode}(CT, sk, x)$ processes as follows, except for the root node and its direct children:
For all the children nodes z of x, the algorithm computes $F_z = \texttt{DecryptNode}(CT, sk, z)$. Let S_x be an arbitrary k_x-sized set of child nodes z where $F_z \neq \perp$. Recall that if there are no such existing set then the node is not satisfied and the algorithm returns $\perp$. In addition, we denote by $i = \texttt{index}(z)$ and $S'_x = \{\texttt{index}(z), z \in S_x\}$. In the sequel, the aggregator computes F_x, defined as:

$$F_x = \prod_{z\in S_x} F_x^{\delta_{i,S'_x}(0)} = \hat{e}(g,g)^{rq_x(0)}$$

Finally, S_j derives $A = \hat{e}(g,g)^{rs}$, associated with the root node and decrypts M such that: $\frac{\tilde{C}}{\frac{\hat{e}(C,D)}{A}} = M$

VI. SECURITY DISCUSSION

To prove the efficiency and the security of our scheme, a security analysis is presented in this section.

A. Correctness

The correctness of the proposed scheme relies on Theorem 6.1, based on the correctness of `aggregate` and `decrypt` algorithms which are defined w.r.t. the multiplication function.

Theorem 6.1: **Correctness –** The proposed scheme is correct if for all security parameter κ, all universe descriptions $\mathbb{S}$, all $(\texttt{pp}, msk) \in \texttt{setup}(\kappa)$, all $\mathcal{S} \subseteq \mathbb{S}$, all $(m, M) \in \mathcal{M}$ (i.e; $\mathcal{M}$ is the message space), all $(\gamma, \Gamma) \in \mathcal{G}$ (i.e; $\mathcal{G}$ is the access structure space), all $sk \in \texttt{keygen}(\texttt{pp}, msk, \mathcal{S})$, all $(C, (s_i, \gamma_i)) \in \texttt{accshare}(\texttt{pp})$ and all $CT_i \in \texttt{encpart}(\gamma_i, s_i, m_i)$, then $\texttt{aggregate}(\texttt{pp}, f, \{CT_i\}_{i\in[1,n]}) = CT$, where n is the number of received ciphertexts, if $\mathcal{S}$ satisfies Γ, such that $\Gamma(\mathcal{S}) = 1$, then the decryption algorithm $\texttt{decrypt}(\texttt{pp}, CT, sk)$ outputs M.

Proof: Upon receiving n different encrypted data chunks for the participating devices $\{CT_i\}_{i\in[1,n]} = \{\tilde{C}_i, \forall y_j \in Y_i, j \in [1, |Y_i|] : C_{i,j}, C_{i,j}\}_{i\in[1,n]}$, the aggregating entity S_j executes the `aggregate` algorithm, with respect to f, defined by GM. The aggregation process mainly relies on the computation of $\tilde{C}$, based on the received $\{\tilde{C}_i\}_{i\in[1,n]}$. That is, the computation of $\tilde{C}$ is performed as follows:

$$\tilde{C} = \prod_{i\in[1,n]} \tilde{C}_i = \prod_{i=1}^{n} m_i \cdot \hat{e}(g,g)^{\alpha \sum_{i=1}^{n} s_i} = M.\hat{e}(g,g)^{\alpha s}$$

In the sequel, the global ciphertext CT is defined as follows:

$$CT = \{\tilde{C}, C, \forall y : C_y, C'_y\}$$

where y is a leaf node of the global access structure Γ.

Upon building the global ciphertext and receiving the secret keys associated to $\mathcal{S}$ (i.e; $\mathcal{S}$ is the set of attributes of the aggregating entity) from AA, the aggregating entity can decrypt CT, if his credentials satisfy the access structure (i.e; $\Gamma(\mathcal{S}) = 1$). Our decryption algorithm mainly relies on the Bethencourt et al. construction [3], w.r.t. the `DecryptNode` function, executed for each access-tree node. As such, the `decrypt` algorithm outputs the resulting message M, as follows:

$$\frac{\tilde{C}}{\frac{\hat{e}(C,D)}{A}} = \frac{M \cdot \hat{e}(g,g)^{\alpha s}}{\frac{\hat{e}(h^s, g^{(\alpha+r)/\beta})}{\hat{e}(g,g)^{rs}}} = \frac{M \cdot \hat{e}(g,g)^{\alpha s}}{\frac{\hat{e}(g,g)^{\alpha s}\hat{e}(g,g)^{rs}}{\hat{e}(g,g)^{rs}}} = M$$

As such, we prove the correctness of our proposed construction, with respect to Theorem 6.1. ∎

B. Confidentiality

To ensure efficient multi-user encryption scheme while enhancing fine-grained access control, our construction mainly relies on the CP-ABE scheme proposed by Bethencourt et al. [3]. As such, the data confidentiality preservation is tightly related to the security of this used encryption algorithm.

Theorem 6.2: The proposed multi-user encryption scheme ensures the secrecy of both encrypted data chunks and aggregated data contents.

Sketch of Proof — the proof of Theorem 6.2 is twofold. First, the secrecy of encrypted data contents depends on the security of the attribute based encryption algorithm used to encipher data chunks provided by IoT devices. Thus, our proposed scheme inherits the indistinguishability property from [3], such that if a malicious adversary has some information about the plaintext, it should not learn about the ciphertext. This security notion requires the computational impossibility to distinguish between two messages chosen by the adversary. Note that in ABE schemes, the adversary may lead an attack against the indistinguishability property either on his own or through a collusion attack. Indeed, similar to [3], each private key element contains a random value r related to each participating entity, which prevents colluding users to override their rights and successfully perform a collusion attack. In addition, sub-access encrypting predicates used to encrypt data chunks are not communicated to the aggregating gateway S_j. As a consequence, this latter cannot deduce data chunk content, even by conducting a brute force attack, because of the use of $C = h^s = h^{\sum_{i=1}^{n} s_i}$. In fact, this value is published by the IoT trusted device, where a malicious aggregator needs to have each single $C_i = h^{s_i}$ and the corresponding sub-access tree γ_i to be able to conduct such attack.

Second, the secrecy of resulting aggregated contents depends on the consistency of the aggregating algorithm agg, such that aggregated data contents are only accessed by the authorized aggregator. Indeed, the general enciphering access predicate is published by the system administrator to the involved aggregating entities. As such, thanks to the use of attribute based encryption, data are only accessed by entities whose attributes satisfy the access policy.

C. Privacy

Theorem 6.3: The proposed scheme is private against both malicious and honest but curious adversaries.

Sketch of Proof — The proof of Theorem 6.3 is twofold. First, it considers the case of malicious users against the privacy of contents. That is, a malicious adversary tries to get access to a single encrypted data chunk. Obviously, this sub-case is related to the confidentiality requirement (cf. Theorem 6.2). Recall that a malicious aggregator cannot deduce data chunks even by conducting a brute force attack, because of the use of a general access tree Γ and a single ciphertext element C. Second, it considers the case of honest but curious adversaries against the privacy of devices. That is, a curious aggregating gateway attempts to distinguish between two legitimate enciphering devices U_1 and U_2, trying to link the encrypted data related to each device. Mainly, our scheme inherits this privacy-preserving property from the used attribute-based encryption mechanism [3]. That is, the encryption of a message relies on a random secret $s = \sum_{i=1}^{n} s_i$, thus a non-identifying enciphering secret, contrary to traditional public key encryption mechanisms. In our proposed multi-user encryption scheme, each device uses a different random s_i to encrypt the sensed message m_i. Thus, our multi-user encrypted access scheme ensures the privacy of users, against honest but curious adversaries.

VII. Performances Analysis

Table I presents a comparison between our proposal and most closely related schemes, w.r.t. functional requirements.

It is worth noticing from Table I that compared to the different presented constructions [14], [3], [18], [17], [11], our proposed scheme provides the multi-user ciphertext-policy encryption feature, relying on general access structures. That is, as stated in section III, although the KP-ABE scheme offers fine-grained access control feature, it has one main disadvantage. Indeed, data owners cannot decide on who has access to their encrypted data, except by their choice of descriptive attributes for the data, since the access policy is embedded in the user private keys. As a result, 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. The data owners can now authorize who can have access on their encrypted data. While constructions proposed in [14], [3] and [11] do not support the aggregation feature, schemes presented in [18] and [17] permit an assistant trusted node to distribute the encryption process among a set of devices, thus allowing it to generate the resulting ciphertext. As such, the aggregating node is mainly the aggregating node, as aggregation cannot be performed by any external entity, unlike our proposed scheme.

The processing and communication costs introduced by the proposed approach are considerably optimized, where the number of polynomials, that have to be assigned to each gate of an access tree, thanks to the use of sub-access trees associated to each participating device, similarly as presented in [18].

TABLE I
COMPARISON BETWEEN OUR SCHEME AND MOST CLOSELY RELATED SCHEMES

scheme	type	access policy	support of multiple encryptors	ciphertext-size	computation cost at the user side	computation cost at the aggregator side
[14]	KP-ABE	threshold	×	$t\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$\gamma_M + (1 + Y_\Gamma)\gamma_E$	×
[3]	CP-ABE	general tree	×	$(2Y_\Gamma + 1)\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$\gamma_M + 2(2 + Y_\Gamma)\gamma_E$	×
[18]	CP-ABE	general tree	×	$(2Y_\Gamma + 1)\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$\gamma_M + 2(2 + Y_{\gamma_i})\gamma_E$	$(2Y_\Gamma + n)\gamma_M$
[17]	KP-ABE	threshold	×	$t(\|\mathbb{G}\| + \|\mathbb{G}_T\|)$	$2Y_{\gamma_i}\gamma_E$	$(1 + n)\gamma_M$
[11]	CP-ABE	AND-gates	×	$3\|\mathbb{G}\| + L$	$Y_\Gamma\gamma_E + 3(\gamma_E + \gamma_M)$	×
ours	CP-ABE	general tree	✓	$(2Y_{\gamma_i} + 1)\|\mathbb{G}\| + \|\mathbb{G}_T\|$	$\gamma_M + 2Y_{\gamma_i}\gamma_E$	$n\gamma_M$

Note: ✓ and × indicate that the requirement is achieved or not, respectively; Y_Γ is the number of attributes in the access policy Γ and Y_{γ_i} is the number of attributes in the access policy γ_i where $\mathbb{U}_{i=[1,n]}\gamma_i = \Gamma$;$|\mathbb{G}|$ and $|\mathbb{G}_T|$ are the size of an element of $\mathbb{G}$ and $\mathbb{G}_T$, respectively; L is the length of the message and t is the threshold value; γ_E and γ_M represent the computation cost exponentiation and multiplication computation costs respectively.

For the `aggregate` algorithm, our construction presents an interesting computation cost $n\gamma_M$, compared to most closely related schemes, namely [18], [17]. For more realistic performance results of our scheme, we conducted some experiments, for several exponentiation and multiplication operations on an Intel *E5-1650-v3* 6 cores. Our measurements show that exponentiations and multiplications take about 1.2 ms and 0.5 ms, respectively.

In addition, referring to the `cpabe` toolkit[2] [3], the computation costs of the key generation and encryption algorithms are mainly depending on the number of involved attributes. The `cpabe` toolkit provides a set of programs implementing CP-ABE schemes, using the PBC library[3]. The code is split into two packages, `libbswabe` (i.e; a library implementing the core cryptographic operations) and `cpabe` (i.e; higher level functions and user interface). For instance, the encryption algorithm takes about 1.5 second relying on an access tree containing around 60 attributes [3].

VIII. POSSIBLE IMPROVEMENTS

The proposed mechanism permits a set of collaborating resource-constrained devices, with the assisting trusted IoT node, to encrypt different sensed data chunks, thus enabling an external gateway to decrypt the set of aggregated data, w.r.t. a general access tree. The assistant node is set up to support IoT devices. However, our multi-user encryption scheme may be deployed for several different settings, namely for multi-fog cooperative applications, where each fog network holds a different data chunk and has to cooperate with other fogs' nodes to encrypt their related data. As such, if a cloud gateway satisfies the general access structure, it can only decipher the aggregated data content.

For this purpose, we extend our proposed scheme, to ensure a collaborative multi-user encryption, that is the set of involved fogs' nodes need to cooperate for deriving the enciphering secret s, based on the key distribution algorithm, introduced in [4]. The different algorithms are defined as follows:

- `setup` – the `setup` algorithm first selects leveled 4-linear maps [8]. In fact, it defines two symmetric pairing functions $\hat{e}_1$ and $\hat{e}_2$, such that $\hat{e}_1 : \mathbb{G}_1 \times \mathbb{G}_1 \rightarrow \mathbb{G}_2$ and $\hat{e}_2 : \mathbb{G}_2 \times \mathbb{G}_2 \rightarrow \mathbb{G}_3$, where $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_3$ are three multiplicative groups of prime order p. It also selects three random generators g_1 and g_2 of $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively. Then, it selects at random two integers $\alpha, \beta \in \mathbb{Z}$. Let $h := {g_1}^\beta \in \mathbb{G}_1$ and $\mathcal{H}$ be a cryptographic hash function, such that $\mathcal{H} : \mathbb{S} \rightarrow \mathbb{G}_1$. The global parameters pp are defined as follows:

$$\mathtt{pp} = \{\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_3, \hat{e}_1, \hat{e}_2, p, g_1, g_2, h, \hat{e}_1(g_1, g_1)^\alpha, \mathcal{H}\}$$

  The master secret key of AA is $msk = \{\beta, {g_1}^\alpha\}$.
- `keygen` – the private key sk contains a new key element, referred to as $\tilde{D}$, and is defined as follows:

$$sk = \{\tilde{D} = {g_1}^{\alpha(1+r)/\beta},$$
$$\forall a_j \in \mathcal{S}, D_j = {g_1}^r \cdot \mathcal{H}(a_j)^{r_j}, D'_j = {g_1}^{r_j}\}$$

- `accshare` – this algorithm is performed by GM. It defines the global access structure Γ, the aggregating function f and the access sub-tree γ_i associated with each participating fog node U_i, such that $\{(U_i, \gamma_i)\}$, where $i \in [1, n]$ and $\Gamma = \cup_{i\in[1,n]}\gamma_i$.
- `secgenpart` – this algorithm is mainly based on [4] construction. It is executed by each participating fog node U_i, in order to derive an element $\mathcal{E}_i$ of his partial secret s_i, needed to compute the global shared enciphering secret s. For this purpose, each participant U_i derives a secret s_i, computes and shares two public elements z_i and μ_i defined as $z_i = h^{s_i}$ and $\mu_i = \hat{e}_1(g_1, g_1)^{\alpha s_i}$, respectively. Afterwards, based on the published elements of the remaining participants, each user U_i calculates and publishes his partial enciphering element $\mathcal{E}_i$ defined as the couple $\mathcal{E}_i = (\mathcal{X}_i, \mathcal{Y}_i) = ([\frac{z_{i+1}}{z_{i-1}}]^{s_i}, [\frac{\mu_{i+1}}{\mu_{i-1}}]^{s_i})$, required to derive the global secret element s.
- `encpart` – this algorithm is run by each U_i, in order to derive a partial ciphertext CT_i for a message m_i, w.r.t. γ_i and the aggregating function f. To do so, each U_i first computes two elements (C, K) associated to the global enciphering secret s, based on the set of shares $\{\mathcal{E}_j\}$, where $j \in [1, n], j \neq i$. That is, each participant calculates C relying on the set of shares $\{\mathcal{X}_j\}_{j\in[1,n],j\neq i}$, such that $C = h^s = h^{s_1s_2+s_2s_3+\cdots+s_ns_1}$. Similarly, U_i computes K relying on the set of shares $\{\mathcal{Y}_j\}_{j\in[1,n],j\neq i}$, such that $K = \hat{e}_2(g_2, g_2)^{\alpha s}$ is derived as follows:

$$\hat{e}_2(g_2, \hat{e}_1(g_1, g_1)^{\alpha s}) = \hat{e}_2(g_2, \hat{e}_1(g_1, g_1)^{\alpha(s_1s_2+\cdots+s_ns_1)})$$

  Then, U_i proceeds similarly as explained in section V. That is, U_i encrypts the message m_i with respect to γ_i

[2]http://acsc.cs.utexas.edu/cpabe/index.html
[3]https://crypto.stanford.edu/pbc/

and outputs the partial ciphertext CT_i such that:

$$CT_i = \{\tilde{C}_i, C, \forall y_j \in Y_i, j \in [1, |Y_i|] : C_{i,j}, C'_{i,j}\}$$

where C value is similarly computed and sent by all participants. As this ciphertext's element depends on the enciphering secret s, it permits to detect whether there is an inconsistency among the different received values, before proceeding to the aggregation and decryption processes. $\tilde{C}_i$ depends on f pointed out by GM and it is defined as $\tilde{C}_i = [m_i]^n \cdot K$.

Recall that the partial ciphertext CT_i associated to the fog node U_i does not include γ_i.

- `aggregate` – it outputs a global CT defined as:

$$CT = \{\tilde{C}, C, \forall Y_i \in \mathcal{L}, \forall y_j \in Y_i : C_{i,j}, C'_{i,j}\}$$

where $\tilde{C}$ is defined as follows:

$$\tilde{C} = \prod_{i=1}^{n} [\tilde{C}_i]^{n^{-1}} = \prod_{i=1}^{n} [(m_i)^n]^{n^{-1}} \prod_{i=1}^{n} K^{n^{-1}} = M \cdot K$$

- `decrypt` – this algorithm, executed by the aggregating entity S_j , permits to decrypt the aggregated ciphertext CT, including the global access structure Γ and the private key sk associated with $\mathcal{S}$. It is mainly similar to the proposed construction, detailed in section V. Thus, for leaves and intermediate nodes, the aggregator computes F_x, defined as: $F_x = \hat{e}_1(g_1, g_1)^{rq_x(0)}$.

For the direct children of the root node, this extended construction introduces a new algorithm referred to as `DecRootChild`. Indeed, it relies on the execution of the `DecryptNode` function and is defined such that $\texttt{DecRootChild}(CT, sk, x, \mu_{i-1}) = \hat{e}_2(\mu_{i-1}, \texttt{DecryptNode}(CT, sk, x))$.

If the aggregating entity satisfies a child-root node x, such that $\texttt{index}(x) = i$, then $\texttt{DecRootChild}(CT, sk, x, \mu_{i-1}) = \hat{e}_2(g_2, g_2)^{\alpha r s_i s_{i-1}}$.

If the set of attributes $\mathcal{S}$ satisfies the global access structure Γ, then the aggregating entity computes a quantity A, as follows:

$$\begin{aligned} A &= \prod_{i\in[1,n], x\in S_r} \texttt{DecRootChild}(CT, sk, x, \mu_{i-1}) \\ &= \prod_{i\in[1,n], x\in S_r} \hat{e}_2(g_2, g_2)^{\alpha r s_i s_{i-1}} \\ &= \hat{e}_2(g_2, g_2)^{\alpha r \sum_{i\in[1,n]} s_i s_{i-1}} \\ &= \hat{e}_2(g_2, g_2)^{\alpha r s} \end{aligned}$$

Finally, the aggregating entity S_j decrypts M as follows:

$$\begin{aligned} \frac{\tilde{C}}{\frac{\hat{e}_2(g_2, \hat{e}_1(C, \tilde{D}))}{A}} &= \frac{M \cdot K}{\frac{\hat{e}_2(g_2, \hat{e}_1(h^s, g_1{}^{\alpha(1+r)/\beta}))}{\hat{e}_2(g_2, g_2)^{\alpha r s}}} \\ &= \frac{M \cdot \hat{e}_2(g_2, g_2)^{\alpha s}}{\frac{\hat{e}_2(g_2, \hat{e}_1(g_1{}^{\beta s}, g_1{}^{\alpha(1+r)/\beta}))}{\hat{e}_2(g_2, g_2)^{\alpha r s}}} \\ &= \frac{M \cdot \hat{e}_2(g_2, g_2)^{\alpha s}}{\frac{\hat{e}_2(g_2, \hat{e}_1(g_1, g_1)^{s\alpha(1+r)})}{\hat{e}_2(g_2, g_2)^{\alpha r s}}} = M \end{aligned}$$

IX. Conclusion

The fast growing and the ubiquity level inflation of Internet of Things leads us to propose a secure and privacy preserving multi-user encryption scheme for fine-grained access control adapted for Cloud-assisted IoT applications.

Our construction takes advantage of the CP-ABE scheme to obtain a flexible and fine grained access control scheme. It enables an aggregating entity to collect sensory data from different devices and decrypt them if it satisfies a general access structure. The proposed cooperative scheme does not reveal any information about the enciphered single data chunks, thus ensuring the privacy of involved entities. Furthermore, it is proved to be suitable for resource-constrained devices based on a theoretical performance analysis.

References

[1] A. Beimel. Secret-sharing schemes: A survey. IWCC'11, 2011.

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

[3] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *Proceedings of the 2007 IEEE Symposium on Security and Privacy*, pages 321–334, Washington, DC, USA, 2007.

[4] M. Burmester and Y. Desmedt. A secure and scalable group key exchange system. *Inf. Process. Lett.*, 94(3), May 2005.

[5] A. Carlin, M. Hammoudeh, and O. Aldabbas. Defence for distributed denial of service attacks in cloud computing. *Procedia Computer Science*, 73:490–497, 2015.

[6] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*, CCS '06, 2006.

[7] M. Hammoudeh, F. Al-Fayez, H. Lloyd, R. Newman, B. Adebisi, A. Bounceur, and A. Abuarqoub. A wireless sensor network border monitoring system: Deployment issues and routing protocols. *IEEE Sensors Journal*, 2017.

[8] S. Hohenberger, A. Sahai, and B. Waters. Full domain hash from (leveled) multilinear maps and identity-based aggregate signatures. In *Advances in Cryptology–CRYPTO 2013*, pages 494–512. Springer, 2013.

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

[10] K. T. Nguyen, M. Laurent, and N. Oualha. Survey on secure communication protocols for the internet of things. *Ad Hoc Netw.*, pages 17–31, Sept. 2015.

[11] V. Odelu, A. K. Das, M. K. Khan, K.-K. R. Choo, and M. Jo. Expressive cp-abe scheme for mobile devices in iot satisfying constant-size keys and ciphertexts. *IEEE Access*, 5:3273–3283, 2017.

[12] N. Oualha and K. T. Nguyen. Lightweight attribute-based encryption for the internet of things. In *Computer Communication and Networks (ICCCN), 2016 25th International Conference on*. IEEE, 2016.

[13] R. Roman, P. Najera, and J. Lopez. Securing the internet of things. *Computer*, 44(9):51–58, 2011.

[14] A. Sahai and B. Waters. Fuzzy identity-based encryption. In *Proceedings of the 24th Annual International Conference on Theory and Applications of Cryptographic Techniques*, EUROCRYPT'05, pages 457–473, 2005.

[15] A. Shamir. Identity-based cryptosystems and signature schemes. In *Proceedings of CRYPTO 84 on Advances in Cryptology*, pages 47–53, 1985.

[16] M. Torchia, M. Kumar, and V. Turner. Worldwide semiannual internet of things spending guide. *IDC (International Data Corporation) June*, 2017.

[17] L. Touati and Y. Challal. Collaborative kp-abe for cloud-based internet of things applications. In *Communications (ICC), 2016 IEEE International Conference on*, pages 1–7. IEEE, 2016.

[18] L. Touati, Y. Challal, and A. Bouabdallah. C-cp-abe: Cooperative ciphertext policy attribute-based encryption for the internet of things. In *Advanced Networking Distributed Systems and Applications (INDS), 2014 International Conference on*, pages 64–69. IEEE, 2014.