Collaborative KP-ABE for Cloud-Based Internet of Things Applications

Lyes Touati, Yacine Challal

Collaborative KP-ABE for Cloud-Based Internet of Things Applications

Lyes Touati
Sorbonne universités,
Université de Technologie de Compiègne,
CNRS, Heudiasyc UMR 7253,
CS 60 319, 60 203 Compiègne cedex. France
Email: lyes.touati@hds.utc.fr

Yacine Challal
Centre de Recherche sur l'Information
Scientifique et Technique CERIST,
05 Rue des Frères Aissou, Ben Aknoun
Algiers, Algeria
Email: ychallal@cerist.dz

***Abstract*—KP-ABE mechanism emerges as one of the most suitable security scheme for asymmetric encryption. It has been widely used to implement access control solutions. However, due to its expensive overhead, it is difficult to consider this cryptographic scheme in resource-limited networks, such as the IoT. As the cloud has become a key infrastructural support for IoT applications, it is interesting to exploit cloud resources to perform heavy operations. In this paper, a collaborative variant of KP-ABE named C-KP-ABE for cloud-based IoT applications is proposed. Our proposal is based on the use of computing power and storage capacities of cloud servers and trusted assistant nodes to run heavy operations. A performance analysis is conducted to show the effectiveness of the proposed solution.**

***Index Terms*—Internet of Things; Cloud Computing; Access Control; KP-ABE;**

I. Introduction

The Internet of Things (IoT) is a novel breakthrough paradigm that gains more and more ground. It allows spreading the Internet to the real world through the connection of physical objects (sensors, actuators, RFID tags, etc.). With this vision, the future Internet will enable a ubiquitous environment where heterogeneous objects can communicate with each other and with users to achieve common goals [1].

The Internet of Things becomes so responded that it is already integrated to the cloud. The large quantity of data objects could generate and the requirements in terms of processing to meet user requests are reasons that justify this approach. However, as objects may need to send sensitive data to the cloud, security measures have to be considered to control access to data. Those measures have to be suitable to cope with limited resources of objects.

Key-Policy Attribute-Based Encryption (KP-ABE) is a powerful encryption scheme which has been widely used in access control systems [2] [3] [4]. However, KP-ABE produces an heavy overhead during the encryption process as it requires a number of exponentiation operation as much as the number of attributes with which the data will be encrypted. This fact is behind the difficulty to use KP-ABE in limited-resource environments like IoT.

Y. Challal is associate professor at Ecole Nationale Supérieure d'Informatique (ESI, Algiers, Algeria). He is member of Systems Design Methods lab. (LMCS)

In this paper we present an approach to apply the KP-ABE scheme for IoT applications based on the cloud. Our solution takes advantage of the capacity of cloud servers to perform heavy operations of KP-ABE.

The rest of this paper is organized as follows. Section II discusses related works. A background of KP-ABE operations is presented in section III. Section IV introduces our approach. A performance analysis of our proposal is presented in section V. Finally, section VI concludes this paper and draws future works.

II. Related Works

There are two main approaches of Attribute-Based Encryption (ABE) (KP-ABE [5] and CP-ABE [6]), both are known to be complex and expensive in terms of energy consumption and computation capability especially for encryption and key generation primitives. This heavy overhead prevents it to be applied in Internet of Things applications. Indeed, most of IoT devices are resource-constrained with limited energy capacities and are not able to support heavy computations. key generation and encryption primitives for both CP-ABE and KP-ABE induce a very heavy overhead, as these two primitives require to do many exponentiation operations over large numbers. This overhead is the reason why ABE schemes are not ready to be implemented in resource-limited devices.

Giuseppe Bianchi et al. [7] have shown by proof-of-concept implementation that computational complexity and energy toll of state-of-the-art multi-authority CP-ABE [8] scheme is still critical. They have proposed a solution AGREE [7] which exploits energy harvesting capabilities of sensors to pre-compute and cache suitably chosen CP-ABE encrypted keys, so as to minimize the need of performing CP-ABE encryptions when no energy from harvesting is available.

In [9] and [10], two online/offline Attribute-Based Encryption solutions are proposed. This approach tackles the computational cost problem of encryption and key generation by splitting each one of these two primitives into two phases: a preparation phase that does the vast majority of the work to encrypt a message or create a secret key. A second phase assembles the ciphertext or the key. These techniques are interesting for mobile devices, since the first phase can be

executed when the device is plugged into a power source, then, when it is unplugged, it can continue executing the second phase without significantly draining the battery.

Another solution was proposed by Lyes Touati et al. in [11] in order to minimize the needed energy during the encryption process. This solution exploits the heterogeneity property of IoT to displace the burden from highly resource-constrained devices to unconstrained trusted ones, it delegates heavy operations of CP-ABE encryption process to neighbor unconstrained nodes.

In this paper, we propose a collaborative variant of KP-ABE named C-KP-ABE for cloud-based IoT applications. Our proposal is based on the use of computing power and storage capacities of cloud servers and trusted assistant nodes to perform heavy operations and save resources of tiny smart objects.

III. Background

In this section, we present some background notions which are necessary for the well understanding of the rest of the paper.

A. Bilinear Maps

Let $\mathbb{G}_0$ and $\mathbb{G}_1$ be two multiplicative cyclic groups of prime order p. Let g be a generator of $\mathbb{G}_0$ and e be a bilinear map, $e : \mathbb{G}_0 \times \mathbb{G}_0 \rightarrow \mathbb{G}_1$. the bilinear map e has the following properties:

1) Bilinearity: for all $u, v \in \mathbb{G}_0$ and $a, b \in \mathbb{Z}_p$, we have $e\left(u^a, v^b\right) = e\left(u, v\right)^{ab}$.
2) Non-degeneracy: $e\left(g, g\right) \neq 1$.

We say that $\mathbb{G}_0$ is a bilinear group if the group operation in $\mathbb{G}_0$ and the bilinear map e are both efficiently computable. Notice that the map e is symmetric since $e\left(g^a, g^b\right) = e\left(g, g\right)^{ab} = e\left(g^b, g^a\right)$.

B. Key-Policy Attribute-Based Encryption

Key-Policy Attribute Based Encryption was firstly introduced by Goyal et al. in [5], where the access policy is specified in users' private key, while the ciphertexts are simply labeled with a set of descriptive attributes. KP-ABE is widely used in data sharing applications where remote servers are semi-trusted, so as the access control mechanism is insured by the encryption mechanism and not by the storage server.

In KP-ABE, data are encrypted over a set of attributes γ which determines the access policy. User's secret key is constructed based on an access tree T. The user is able to decrypt the ciphertext if and only if γ satisfies T.

KP-ABE scheme requires a special entity in the system named **Attribute Authority** that manages the public parameters namely the universe of attributes $\mathcal{U}$. It is also charged to generate the public and the master keys and all users' privates keys.

Figure 1 shows a simple example of application where KP-ABE is used. In the left figure, sensors are deployed in the area where we want to do measures. The sink mote collects all the measures from the sensors and encrypts them under an attributes set γ. Then, the encrypted file is sent to the cloud where it is available via web based application for users. Only those who have a secret key with an access tree satisfied by γ are able to decrypt the encrypted file and access the plaintext.

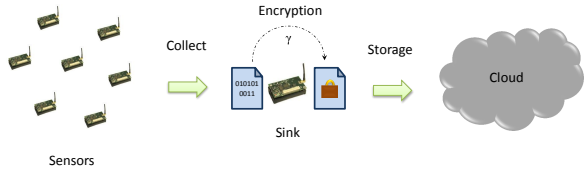

Figure 1: Application Example

Access tree.
The access tree in KP-ABE scheme defines the access scope of a user's secret key. Each non-leaf node of it represents a threshold gate, described by its children and a threshold value. If num_x is the number of children of a node x and k_x is its threshold value, then $0 < k_x \leq num_x$. Each leaf node x of the tree is described by an attribute and a threshold value $k_x = 1$.

Some functions are defined to facilitate working with access trees:

- *parent(x)*: denotes the parent of the node x in the tree.
- *att(x)*: is defined only if x is a leaf node, and denotes the attribute associated with the leaf node x in the tree.
- *index(x)*: denotes the order of the node x between its brothers. The nodes are numbered from 1 to num.

Satisfying an access tree.
Let T be an access tree with root r. Denote by T_x the sub-tree of T rooted at the node x. Hence T is the same as T_r. If a set of attributes γ satisfies the access tree T_x, we denote it as $T_x\left(\gamma\right) = 1$. We compute $T_x\left(\gamma\right)$ recursively as follows. if x is a non-leaf node, evaluate $T_{x'}\left(\gamma\right)$ for all children x' of node x. $T_x\left(\gamma\right)$ returns 1 if and only if at least k_x children return 1. if x is a leaf node, then $T_x\left(\gamma\right)$ returns 1 if and only if $att\left(x\right) \in \gamma$.

Figure 2 illustrates a simple example of an encrypted file with the attributes set $\gamma = \{A, B, C\}$. Three users attempt to decrypt the encrypted file. The first and the third users have secret keys corresponding to the access trees $T_1 \equiv (\text{'}A\text{'}$ **OR** $\text{'}D\text{'})$, $T_3 \equiv (\text{'}D\text{'}$ **OR** $(\text{'}B\text{'}$ **AND** $\text{'}C\text{'})$ respectively. As $T_1(\gamma) = 1$ and $T_3(\gamma) = 1$ (γ satisfies both T_1 and T_3), user 1 and 3 can both decrypt the file. However, the secret key of the second user is constructed over the access tree $T_2 \equiv (\text{'}A\text{'}$ **AND** $\text{'}D\text{'})$, and γ does not satisfy T_2. Therefore, User 2 cannot access the plaintext of the encrypted file.

KP-ABE scheme.
Let $\mathbb{G}_1$ be a bilinear group of prime order p, and let g be a generator of $\mathbb{G}_1$. Let $e : \mathbb{G}_1 \times \mathbb{G}_1 \longrightarrow \mathbb{G}_2$ denote the bilinear map. We also remind the Lagrange coefficient:

$$\forall i \in \mathbb{Z}_p, \forall S \subseteq \mathbb{Z}_p : \Delta_{i,S}(x) = \prod_{j \in S, j \neq i} \frac{x - j}{i - j} \qquad (1)$$

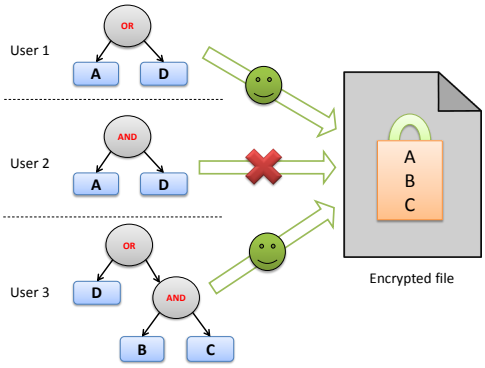

Figure 2: Example KP-ABE

Setup:

This primitive is executed during the bootstrap phase in order to create the Master Key MK and the Public Key PK. It takes no input other than the implicit security parameter.

It starts by defining the universe of attributes $\mathcal{U} = 1, 2, ..., n$. For each attribute $i \in \mathcal{U}$, a number t_i randomly chosen from $\mathbb{Z}_p$. The primitive chooses also a random number y from $\mathbb{Z}_p$. The Public Key PK is published as:

$$PK = (T_1 = g^{t_1}, \ldots, T_{|\mathcal{U}|} = g^{t_{|\mathcal{U}|}}, Y = e(g,g)^y). \quad (2)$$

and the Master Key MK is:

$$MK = (t_1, \ldots, t_{|\mathcal{U}|}, y). \quad (3)$$

The public key PK is published and shared with all entities in the system while the master key MK is kept secret.

Encryption Primitive:

To encrypt a message $M \in \mathbb{G}_2$ under a set of attributes γ, choose a random value $s \in \mathbb{Z}_p$ and publish the ciphertext as:

$$E = (\gamma, E' = MY^s, \{E_i = T_i^s\}_{i \in \gamma}) \quad (4)$$

Key Generation Primitive:

The algorithm outputs a secret key SK that is associated with an access tree T. It is run by the Attribute Authority for each user in the system. The key generated is able to decrypt a ciphertext E which is labeled with a set γ of attributes if and only if γ satisfies T ($T(\gamma) = 1$).

As the algorithm steps are beyond the scope of this paper, we do not provide more details about the algorithm. However, the reader can get more details about it in [5].

Decryption Primitive:

The decryption algorithm takes as input a ciphertext E and user's secret key SK. It outputs an element from $\mathbb{G}_2$ if $T(\gamma) = 1$, where T represents the access tree associated to SK, and γ is the list of attributes in E. In this case, the element from $\mathbb{G}_2$ represents the original message M. otherwise, it outputs $\perp$ which means that SK is not able to decrypt E.

More details about the algorithm can be found in [5].

IV. Proposed Solution: Collaborative KP-ABE

A. Motivations

The main source of overhead in the KP-ABE scheme is in the encryption and the generation of secret keys primitives, this is because of the number of exponentiation to be computed. Indeed, there are $|\gamma| + 1$, and $|S|$ exponentiation in the encryption and secret keys generation primitives respectively, where, γ is the attributes set defining the access policy to the ciphertext, and S is the set of leaf nodes in the access tree associated to the secret key.

In this paper, we are interested in the encryption primitive which is usually the most frequently called by resource-constrained nodes. The main idea of our proposal is to exploit computing power and storage capacities of cloud servers, and IoT heterogeneity to delegate the most costly operations (namely exponentiation) to less constrained nodes in the neighborhood of the encryptor.

B. Network Model

It is well known that the IoT infrastructure is usually heterogeneous i.e. the objects in the network do not all have the same capabilities in terms of computing power and energy resources. Indeed, we can distinguish three categories of nodes:

- *Resource-constrained devices*, unable to support the computational cost of the KP-ABE encrypt operation, while nevertheless must send sensitive data (e.g. sensor nodes).
- *Unconstrained devices*, therefore able to perform costly operations. These nodes may either be dedicated assisting servers or nodes belonging to the same local infrastructure, though being less impacted by energy constraints (e.g. having energy harvesting capability, line-powered devices).
- *Remote servers (Cloud)*, characterized by high computing power and high storage capacities and have a functional role in the system, they are intended to store encrypted messages received from system's objects and make them available to any request from any entity of the system.

C. Assumptions

1) For each resource-constrained device there are at least two trusted unconstrained devices in its neighborhood.
2) Every resource-constrained device shares pairwise keys with two or more unconstrained devices in its neighborhood. These keys may have been generated during a specific bootstrapping phase
3) Every object in the system shares pairwise key with the remote server.

D. Notation

Table I presents the list of notations used in the paper to describe our Collaborative Key-Policy Attribute-Based Encryption (C-KP-ABE) solution.

Table I: Summary of notations.

Notation	Description
MK	Master Key generated by the Attribute Authority
PK	Public Key generated by the Attribute Authority
$\mathcal{U}$	The universe of attributes
E	Ciphertext generated by the encryption process
M	The message to be encrypted
S	Set of leaf nodes in the access tree of a user
γ	Set of attributes associated to the ciphertext
K_{AS}	Shared symmetric key between the device A (the encryptor) and the remote server
K_{AP_j}	Shared symmetric key between device A and the j^{th} assistant node
K_{SP_j}	Shared symmetric key between the remote server and the j^{th} assistant node
p	Number of sought trusted assistant nodes during encryption process
n	Number of attributes used for encryption
p_{max}	Maximum number of assistant nodes without losing in communication energy

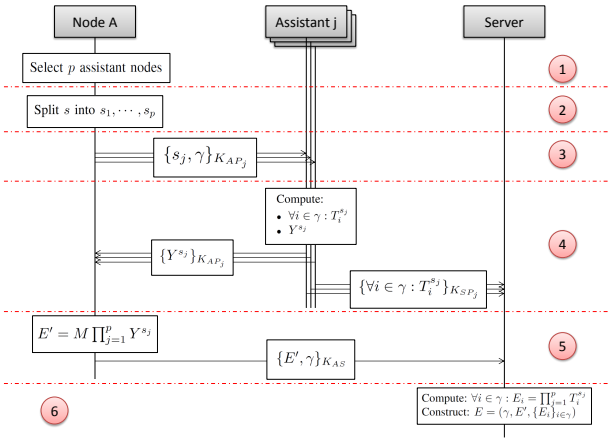

Figure 3: C-KP-ABE Scheme

E. Per-phase Collaborative Encryption Primitive

Let A be a constrained node. **Node A** aims to encrypt a data under a list of attributes γ and send the result to the **Cloud Server**. The server stores encrypted data sent by IoT devices. During the encryption process, node A is helped by a set of trusted **Assistant Nodes**. These assistant nodes are selected from the neighborhood of node A. They execute the exponentiation operations instead of Node A.

Figure 3 shows node A in the left, cloud server on the right, and assistant nodes in the middle. It illustrates the different steps of our Collaborative KP-ABE encryption process. These steps are detailed bellow.

1) Phase 1: The encryptor (Device A in figure 3) selects p trusted nodes among those situated in its neighborhood, these nodes (assistant nodes) will assist the device A during the encryption process. Notice that the existence of such trusted nodes is assumed in the network model (section IV-C).
2) Phase 2: First, the encryptor defines the set of attributes γ that will be associated to the ciphertext. Then, it chooses a random value $s \in \mathbb{Z}_p$ and split it into p values s_j such that the sum of all these values gives s (formula 5).

$$s = \sum_{j=1}^{p} s_j. \tag{5}$$

3) Phase 3: The encryptor sends each s_j to an assistant node j along with the attributes set γ, all encrypted with the shared symmetric key K_{AP_j} with the latter. the value of Y doesn't require to be sent because it is a part of the public key PK, so all the assistant nodes already know it.
4) Phase 4: After receiving s_j, the j^{th} assistant node computes Y^{s_j}, and for all $i \in \gamma : T_i^{s_j}$. Then, it sends back Y^{s_j} to the encryptor (device A), and hands over all $\{T_i^{s_j}\}_{i \in \gamma}$ to the remote server after encryption with the shared key.
5) Phase 5: When the encryptor receives the interim results from assistant nodes, it computes E' as follows:

$$\begin{aligned} E' &= M \prod_{j=1}^{p} Y^{s_j} \\ &= M Y^{\sum_{j=1}^{p} s_j} \\ &= M Y^{s}. \end{aligned} \tag{6}$$

After that, the encryptor sends this result along with the attributes set γ to the remote server all encrypted by the symmetric shared key K_{AS} with the latter.
6) Phase 6: The remote server receives interim results from assistant nodes and computes $\{T_i^s\}_{i \in \gamma}$ elements.

$$\begin{aligned} \forall i \in \gamma : E_i &= \prod_{j=1}^{p} T_i^{s_j} \\ &= T_i^{\sum_{j=1}^{p} s_j} \\ &= T_i^{s} \end{aligned} \tag{7}$$

It receives also from the encryptor the attributes set γ and E'. It constructs the ciphertext as follows:

$$E = (\gamma, E', \forall i \in \gamma : E_i = T_i^s) \tag{8}$$

The ciphertext E will henceforth be available by the cloud server to whoever requests it.

V. Analysis

A. Security Analysis

To decrypt a ciphertext E, a user must have a valid secret key SK whose corresponding access tree T satisfies the attributes list γ associated with E. The authorized user recovers the value of Y^s using her/his secret key SK and E_i. After that, she/he can get M by a simple division of E' by Y^s [5]. Therefore, an unauthorized user can recover the plaintext M if only she/he knows Y^s or even the random number s

used to encrypt the data. Otherwise she/he must have a valid secret key to do so.

An unauthorized user who eavesdrops the communications cannot get any information about Y^s or s during the collaboration of the encryptor, assistant nodes, and the storage server. Because all the communications are encrypted using symmetric shared keys (Assumptions 2 and 3).

Assumption 1 presented in section IV-C ensures that assistant nodes will not disclose any information exchanged with the encryptor or even collide between them to compute s or Y^s using s_i or Y^{s_i} $(i \in 1, \cdots, p)$ respectively. Indeed, assistant nodes are assumed to be trusted. Assumption 1 also ensures that assistant nodes will not give erroneous results or abstain from participating to the encryption process.

The remote server also cannot recover the plaintext of the encrypted data. The information received from the assistant nodes and the encryptor do not allow computing s or Y^s.

To prevent our solution from *replay attacks* we can use a nonce at the beginning of the encryption process. So as, attackers cannot reuse old communications.

B. Performance analysis

Table II shows a comparison between our solutions and the native KP-ABE in terms of number of multiplications and exponentiation operations.

Table II: Computation comparison

		Number of Multiplications	Number of Exponentiations
KP-ABE	Device A	1	$\|\gamma\|+1$
	Remote server	0	0
	Assistant node	-	-
C-KP-ABE	Device A	p	0
	Remote server	$\|\gamma\|(p-1)$	0
	Assistant node	0	$\|\gamma\|+1$

We notice that our solution C-KP-ABE removes exponentiation operations from the encryptor (Device A) against a increase of the number of multiplications. The latter increases proportionally with the number of assistant nodes p, but its impact remains negligible compared to the high number of exponentiation operations in the native KP-ABE (Table IV, Figure V).

Experiment settings.
These simulations were made on a virtual machine 32 bits ubuntu 12.04 LTS with 1GB of RAM. The processor used is Intel® Core™ i7-3540M CPU @ 3.00GHz. We used PBC-0.5.14 and GMP-6.0.0 to implement our solution. In this implementation, we used pairing parameters in the file *"a1.param"* to initialize our pairing.

Sizes of all elements used in KP-ABE are given in Table III. We recall that these sizes are specific to the parameters in the file *"a1.param"*. We add to this table the size of attributes identifier that we set to 4 bytes. This allows to manage up to $2^{(32)} = 4.2$ billions of attributes by the Attributes Authority.

Table III: Element Size

Group	Size (bytes)
$\mathbb{G}_1$	264
$\mathbb{G}_2$	264
$\mathbb{Z}_p$	132

The simulation conducted consider only the time required to execute encryption operations. The time required to send assistance requests and receive the responses is not taken into account in these simulations.

Table IV: Execution time of the encryption primitive for both KP-ABE and C-KP-ABE with five assistant nodes ($p = 5$)

Nb. attributes in γ	**KP-ABE** (s)	**Collaborative KP-ABE** (s)
1	0,046522018	0,001412513
10	0,418102258	0,001474124
20	0,853737376	0,001429133
30	1,255189945	0,001462039
40	1,647791114	0,001500966
50	2,115445595	0,001475357
60	2,458539065	0,001500431
70	2,820171402	0,001452353
80	3,303311991	0,001518491
90	3,706833459	0,001486667
100	4,014498504	0,001607614

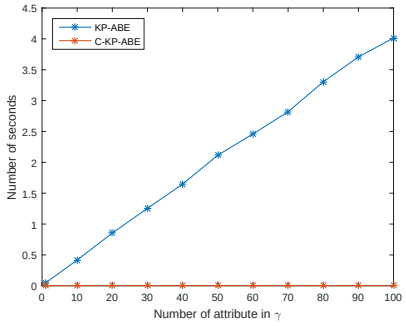

Figure 4: Comparison of the execution time. $p = 5$

In Table IV and Figure 4 we show results of experiments conducted on KP-ABE and C-KP-ABE by varying the number of attributes in γ. In these experiments, we have interested in the time execution of the encryption primitives of both KP-ABE and C-KP-ABE with five (5) assistant nodes. We notice that our C-KP-ABE show best results; the time execution is very negligeable while the curve of KP-ABE grows almost linearly with respect to the number of attributes.

Impact of the number of assistant nodes.
In this simulation, we show the impact of the number of selected assistant nodes on encryption performances. These results indicate the execution time of the encryptor-side. We notice that transmission and reception times are not included, only cryptographic operations execution time counts. The number of attributes is fixed to one hundred $(n = 100)$. The number of assistant nodes varies between 2 and 50.

The results of this simulation are shown in table V and figure 5. We notice that the execution time increases with the number of the sought assistants nodes. However, even with fifty (50) assistant nodes and one hundred attributes used for encryption, the time needed to execute encryptor's operations is less than the one of KP-ABE using one attribute. The reason behind the performance of C-KP-ABE is the elimination of exponentiation operations and delegating them to assistant nodes.

Table V: Variation of the encryption execution time of C-KP-ABE with respect to the number p of assistant nodes sought. $|\gamma| = 100$.

Nb. of assistant nodes	**Encryption time** (s)
2	0.000819221
5	0.001754341
10	0.003837017
15	0.003848409
20	0.008189308
25	0.010148681
30	0.014133818
35	0.016193901
40	0.019061304
45	0.020961044
50	0.022334400

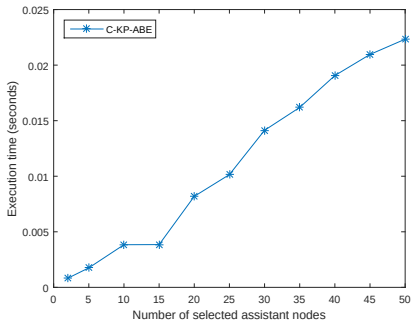

Figure 5: Execution time variation with respect to number of assistant nodes. $n = |\gamma| = 100$.

Communication cost.

In order to compare the communication cost between our solution and the native KP-ABE we adopted the energy model proposed in [12]. The amount of energy needed by the radio to transmit a l-bit message a distance d is given with formula 9.

$$\begin{aligned} E_{Tx}(l,d) &= E_{Tx-elec}(l) + E_{Tx-amp}(l,d) \\ &= lE_{elec} + l\epsilon_{fs}d^2 \end{aligned} \tag{9}$$

Where $E_{Tx-elec}(l)$ represents the electronics energy and $E_{Tx-amp}(l,d)$ is the amplifier energy. d represents the distance between the transmitter and the receiver which is smaller then the threshold d_0 as we have assumed that all assistant nodes are situated in the neighborhood of Node A. d is set to $100m$. The communication energy parameters are set as: $E_{elec} = 50nJ/bit$, $\epsilon_{fs} = 10pJ/bit/m^2$. To receive a message, the radio expends:

$$\begin{aligned} E_{Rx}(l) &= E_{Rx-elec}(l) \\ &= lE_{elec} \end{aligned} \tag{10}$$

Let p be the number of assistant nodes sought for the encryption process. For example, we set $p = 5$ and we compared communication costs of KP-BAE and C-KP-ABE for different sizes of ciphertext in terms of number of attributes ranging from 1 to 50. The results of comparison are summarized in table VI and represented also in Figure 6.

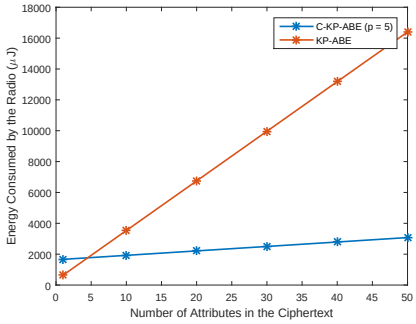

Figure 6: Comparison of Communication Cost (Number of assistant nodes $p = 5$).

When the number of attributes is low (in this example one attribute is used), KP-ABE overcomes our C-KP-ABE in term of communication cost. However, with the growing of the number attributes used for encryption, our solution overcomes widely the original KP-ABE encryption primitive. It is important to recall that these results do not consider the computation cost as well.

Joint impact of the number of assistant nodes p and the number of attributes n.

After this result, we were interested in finding the maximum number of assistant nodes p_{max} that the node A can solicit without losing energy compared to KP-ABE. This number of assistant nodes depends on the number of attributes n used for encryption.

From formulas 9 and 10, $E_{Tx}(1, 100) = 150\ nJ$ represents the energy needed by the radio to send one bit over $100m$, and $E_{Rx}(1) = 50\ nJ$ is the energy for receiving one bit.

Let E_{KP-ABE} and $E_{C-KP-ABE}$ be the communication energy consumed during the encryption process of the original KP-ABE and our Collaborative KP-ABE respectively. Let E_{save} be the energy saving using our solution during the encryption process, E_{save} depends on the number of assistant nodes sought p and the number of attributes in the cipher-text n.

Table VI: Comparison of Communication Cost (Number of assistant nodes $p = 5$).

Nb. of Attributes	Nb. Bytes sent (bytes)		Nb. Bytes received (bytes)		Total Energy (μJ)	
	KP-ABE	C-KP-ABE	KP-ABE	C-KP-ABE	KP-ABE	C-KP-ABE
1	532	948	-	1320	638	1665.6
10	2944	1164	-	1320	3533	1924.8
20	5624	1404	-	1320	6749	2212.8
30	8304	1644	-	1320	9965	2500.8
40	10984	1884	-	1320	13181	2788.8
50	13664	2124	-	1320	16397	3076.8

$$\begin{aligned} E_{save}(p,n) &= E_{KP-ABE}(n) - E_{C-KP-ABE}(p,n) \\ &= E_{Tx}(1,100)\cdot 264n - E_{Tx}(1,100)\cdot p(132+4n) \\ &\quad + E_{Rx}(1)\cdot 264p \\ &= E_{Tx}(1,100)\cdot(264n - p(132+4n)) \\ &\quad - E_{Rx}(1)\cdot 264p \end{aligned} \tag{11}$$

For each value of n ranging from 1 to 100, we try to find the maximum value p_{max} of p so as the energy saving remains positive ($E_{save}(p,n) \geq 0$). These results are shown in Figure 7.

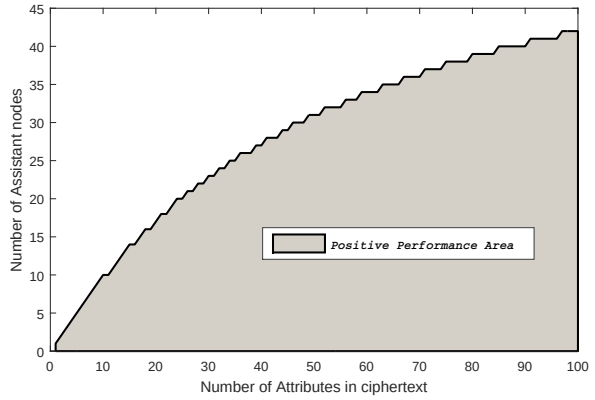

Figure 7: Maximum Number of Authorized Assistant Nodes.

We notice that p_{max} increases with high values of n. This allows to define an area where to choose p without exceeding p_{max} so that the overall energy cost is less than the cost in the original KP-ABE.

VI. Conclusion

In this paper, we have proposed a distributed variant of an encryption technique leveraging the heterogeneous nature of the Internet of Things, in order to implement it in resource-constrained devices. Our solution delegates heavy operations of the encryption process to trusted unconstrained assistant nodes and a cloud server.

As future work, it would be interesting to consider the selfishness problem of assistant nodes. Indeed, some assistant nodes could try to preserve their energy by refusing to assist resource-constrained nodes in their encryption process.

Mobility of devices in IoT is also a tricky problem to address; this would lighten the assumption 1 by assuming only that at least two trusted nodes are available in the neighborhood of the constrained node during a period of time.

VII. Acknowledgment

This work was carried out and funded in the framework of the Labex MS2T. It was supported by the French Government, through the program "Investments for the future" managed by the National Agency for Research (Reference ANR-11-IDEX-0004-02).

References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787 – 2805, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1389128610001568

[2] C. Wang and J. Luo, "An efficient key-policy attribute-based encryption scheme with constant ciphertext length," *Mathematical Problems in Engineering*, vol. 2013, 2013.

[3] Y.-m. Ji, J. Tan, H. Liu, Y.-p. Sun, J.-b. Kang, Z. Kuang, and C. Zhao, "A privacy protection method based on cp-abe and kp-abe for cloud computing," *Journal of Software*, vol. 9, no. 6, pp. 1367–1375, 2014.

[4] S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving secure, scalable, and fine-grained data access control in cloud computing," in *INFOCOM, 2010 Proceedings IEEE*. Ieee, 2010, pp. 1–9.

[5] 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*, ser. CCS '06. New York, NY, USA: ACM, 2006, pp. 89–98.

[6] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *Security and Privacy, 2007. SP '07. IEEE Symposium on*, May 2007, pp. 321–334.

[7] G. Bianchi, A. T. Capossele, C. Petrioli, and D. Spenza, "Agree: Exploiting energy harvesting to support data-centric access control in wsns," *Ad Hoc Netw.*, vol. 11, no. 8, pp. 2625–2636, Nov. 2013.

[8] A. Lewko and B. Waters, "Decentralizing attribute-based encryption," in *Proceedings of the 30th Annual International Conference on Theory and Applications of Cryptographic Techniques: Advances in Cryptology*, ser. EUROCRYPT'11. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 568–588.

[9] S. Hohenberger and B. Waters, "Online/offline attribute-based encryption," in *Public-Key Cryptography - PKC 2014*, ser. Lecture Notes in Computer Science, H. Krawczyk, Ed. Springer Berlin Heidelberg, 2014, vol. 8383, pp. 293–310.

[10] P. Datta, R. Dutta, and S. Mukhopadhyay, "Fully secure online/offline predicate and attribute-based encryption," in *Information Security Practice and Experience*, ser. Lecture Notes in Computer Science, J. Lopez and Y. Wu, Eds. Springer International Publishing, 2015, vol. 9065, pp. 331–345.

[11] 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*, June 2014, pp. 64–69.

[12] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An application-specific protocol architecture for wireless microsensor networks," *Wireless Communications, IEEE Transactions on*, vol. 1, no. 4, pp. 660–670, Oct 2002.