COLLABORATIVE AND FAST DECRYPTION USING FOG COMPUTING AND A HIDDEN ACCESS POLICY

Ahmed Saidi[1], Omar Nouali[2] and Abdelouahab Amira[3]

[123]Department of Computer Security, Research Center for Scientific and Technical Information, Algiers, Algeria

[13]Faculty of Exact Sciences, Universite de Bejaia, 06000 Bejaia, Algeria.

ABSTRACT

Nowadays, IOT (Internet Of Things) devices are everywhere and are used in many domains including e-health, smart-cities, vehicular networks,.. etc. Users use IOT devices like smartphones to access and share data anytime and from anywhere. However, the usage of such devices also introduces many security issues, including in data sharing. For this reason, security mechanisms such as ABE (Attribute-Based Encryption) have been introduced in IOT environments to secure data sharing. Nevertheless, Ciphertext-Policy ABE (CP-ABE) is rather resource intensive both in the encryption and the decryption processes. This makes it unadapted for IOT environments where the devices have limited computing resources and low energy. In addition, in CP-ABE, the privacy of the access policy is not assured because it is sent in clear text along with the cipher-text. To overcome these issues, we propose a new approach based on CP-ABE which uses fog devices. The letters collaborate to reduce the bandwidth, and partially delegates data decryption to these fog devices. It also ensures the privacy of the access policy by adding false attributes to the access policy. We also discuss the security properties and the complexity of our approach. We show that our approach ensures the confidentiality of the data and the privacy of the access policy. The complexity is also improved when compared with existing approaches.

KEYWORDS

Fog Computing, Access Control, Attribute based Encryption, Decryption Outsourcing.

1. INTRODUCTION

Fog computing is an emerging paradigm that extends cloud computing. It acts as an intermediary between the cloud and end devices by bringing processing, storage and networking services closer to these end devices[1]. For example, the processing and the storage of temporary data which are collected by sensors can be delegated to the hospital local servers which act as fog nodes. This architecture allows to reduce the amount of data transferred to the cloud for processing, analysis and storage. As a consequence, the network traffic bandwidth and latency are reduced. This is especially the case of data sharing, which allows users to store their data, access it from anywhere, at anytime, and share it with other users. However, Users lose control over their data when it is outsourced to the Cloud or when it is processed by fog nodes.

Natarajan Meghanathan et al. (Eds) : CSEIT, CMLA, NeTCOM, CIoT, SPM, NCS, WiMoNe, Graph-hoc - 2019
pp. 57-71, 2019. © CS & IT-CSCP 2019 DOI: 10.5121/csit.2019.91305

To address these issues, it is essential to use mechanisms such as encryption and decryption, which allow to secure data sharing.

ABE [2] is a new, efficient and promising encryption/decryption technique that aims to achieve scalable and fine-grained access control. It keeps the encrypted data confidential even when the storage server is untrusted. In ABE, the encryption is based on a set of attributes describing data properties, user properties and properties of the environment, as well as an access structure indicating who can access what. ABE is constructed from an access tree representing a logical expression that combines several attributes via AND and OR operators. There are two main variants of ABE: (1) ABE Key-Policy (KP-ABE) [3] and (2) ABE-Ciphertext-Policy (CPABE) [2]. The KP-ABE, the encrypted data is associated with a set of attributes. Whereas, the key is associated with the access policy. The users can decrypt the data if and only if the attributes in the data satisfy the access policy. On the other hand, in CP-ABE, the attributes are associated with the user's private key and the data is encrypted with the access policy.

Nevertheless, one of the drawbacks of ABE is that the computational cost during in the encryption and decryption phases increases exponentially with the complexity of the access policy. This is a considerable limitation when devices are limited in terms of resources (for example CPU, energy, etc.). Another drawback of ABE is that the access policy is sent in clear text along with the ciphertext. A malicious user can obtain both the ciphertext and the associated access policy. The latter contains some sensitive information (like social security number, name,etc), that can be exploited to compromise the legitimate user's privacy.
In this paper, we propose a new solution based on CP-ABE. Our approach uses fog nodes collaboration and a newly proposed partial decryption approach with a hidden access policy to achieve low computation overhead and achieve secure and fine access control.
The basic idea of our approach is as follow:

(1) We use Fog nodes to offer for fast and more convenient computing services. Moreover, the fog computing provides low-latency communications.

(2) Our scheme delegates the user's attributes and the decryption operation to the fog nodes without revealing the original message, the set of user's attributes or the attributes in the access policies to the fogs. Fog nodes collaborate with each other to help the user decrypt the data. To delegate the decryption operation, the TA (Trusted Authority) creates intermediate keys for the fog nodes using the user's secret key. This intermediate key is used by the fog to partially decrypt the text without revealing which attributes are used in the decryption process.

(3) We add false attributes to hide the access policy. The trusted authority divides the set of attributes over all available fogs. Each fog manages it own set of attributes. When user (Data Owner) creates an access policy, he divides the access policy and adds false attributes to each subtree of the access policy. This operation is performed by taking into account the number of available fog and according to the set of attributes managed by Fog node. In this way, the fog nodes will not be able to deduce which attributes participated in the decryption phase.

Contribution:

The main contributions of this paper are as follow:

- To the best of our knowledge, our work is the first to hide and protect user attributes against fogs nodes in outsourced decryption process phase using fog nodes.

- We present a secure outsourcing and a fast decryption approach by delegating heavy computations from IOT to Fog. This is performed by creating an intermediates key from the user which are used to partially decrypt the ciphertext. This means that the computational decryption complexity of IOT is independent of the number of attributes.

- We divide the set of universal attributes by the set of available fogs so that each fog manages its proper attributes. When the data owner wants to encrypt the data, the access policy is divided according to the attributes that each fog manages.

- We extend our approach by adding false attributes for to each access policy so that fogs nodes cannot deduce the real attributes. In addition, fog nodes are not able to deduce the valid attributes users in the decryption process even if the case where fog nodes are compromised or collude.

- We thoroughly analyze the security properties and the decryption complexity of our proposed scheme.

Paper organization:

The reset of this paper is organized as follows. In Section II, we examine the existing solutions that aim to reduce the encryption and decryption costs. In Section III, we introduce the system and threat models. In Section IV, we give a high-level overview of the proposed scheme. The detailed construction of our ABE based outsourced decryption scheme is given in Section VI. We analyze the security and complexity of our approach in Section V. In section VII, we introduce some typical scenarios where our proposed scheme can be applied. The paper is concluded in Section VIII.

2. RELATED WORKS

Attribute-Based Encryption (CP-ABE) [2] is considered one of the most appropriate technologies for performing fine-grained access control. However, the encryption and decryption processes in this scheme are very complex and time consuming. In order to reduce the cost of encryption and decryption at the user level, several schemes for externalizing the computation were proposed.

Zhou et al. [4] proposed a new CP-ABE scheme, in which the encryption and decryption process is outsourced on external cloud based services. In the encryption process, the authors connect two access structures T1 and T2 to form a single access policy. A root AND node connect these access policies. The first part of the encrypted text is generated by sending T1 to an external encryption service while the second part is computed by the user using T2, where this T2 contains only one attribute. However, one flaw in this approach is that the access policy in this scheme is not hidden.

In their work Touati et al. [5] present a cooperative CP-ABE for the Internet of Things, where the complex operations of the CPABE encryption primitive forced authors to use intermediates Unconstrained devices to outsourced encryption process. The authors assume that unconstrained devices are trusted. In this scheme, the data owner (device A that is a resource constrained device) encrypts the data under access T. During the process; device A is supported by a set of secure assistant devices that perform the exponentiation operation instead of the device A itself. The authors suppose that the intermediate unconstrained devices are trusted, but they do not suggest externalization of the decryption process, Another drawback is that the

access policy is sent in the clear on the network, where the access structure can also contain some sensitive private information.

In [6], the authors propose a new method for outsourcing CP-ABE, namely the EOEB (outsourcing mechanism for the encryption of the ABE encryption policy). The main idea is to reduce encryption costs by delegating the most intensive computations of the encryption phase of the CP-ABE to a semi-trusted party. The authors divide the encryption process in to two phases: the Pre-delegation phase, and the compDelegation phase. Pre-delegation is performed by KDG (Key delegation) which executes the configuration algorithm as in the basic CP-ABE. It also generates a secret delegation key for each data producer (DP) and a list of security parameters. This list is then sent to DG (delegate). Two steps are executed in the compDelagation phase. The first step is executed by DP. In this step, the DP generates the temporal encrypted text CT' which contains the Blinded value s. The second step is executed by DG (delegate) which executes the most expensive computation operation without any knowledge of the secret message M. Nevertheless as in the work of [5], the authors do not propose to outsource the decryption process which consumes IOT energy at the user level and they do not hide the access policy.

Fan et al. [7] proposed an outsourced, secure and verifiable multi-authority access control system called VO-MAACS. In their construction, most encryption and decryption computations are outsourced to Fog devices, and the result can be verified by signed the message. The Fog devices are responsible for the transmission of data. They are also responsible for a part of the computation of encryption and decryption. Fog devices can help data owners to generate some of the encrypted text. They also help DVs to decrypt some of the encrypted text but only when the DV attributes satisfy the access policy. In this proposal, the authors used a secret linear sharing scheme (LSSS) to construct an access policy. Despite this, in their scheme, the access policy is not hidden.

In [8] propose a CP-ABE scheme with a hidden access strategy and fast decryption that improves the decryption efficiency at the user level. The authors also propose a method to hide the access policy by adding false attributes to the access policy which preserves its confidentiality. This method ensures fast decryption and hidden access policy. However, in this scheme even if there is an improvement in energy consumption in the IOTs. The decryption process still is the energy intensive since it is executed at the user level.

In [9] the authors proposed Securely outsourcing multi-authority attribute based encryption with policy hidden for cloud assisted IoT (PHOABE), in this scheme, the attributes in access policy are hidden, and the decryption process is outsourced to the third party. However, the solution is proven selectively secure and even though the decryption process is outsourced the overhead cost at user still important Thus we rely on the work of Wang and Lang [8] where we have modified their scheme for outsourcing of the decryption process to several Clouds, using intermediate keys for partial decryption of the data.

In existing approaches, the policy is not hidden in other works energy intensive and selective privacy methods is used.in other cases, outsourcing is not assured.

3. BACKGROUND

In this section, we present some notation used in this article. Then we illustrate the details of the encryption and decryption process.

A) Preliminaries

1) Composite-Order Bilinear Group: Let G denote an algorithm that takes as input a security parameter and outputs a tuple $(N = p_1p_2p_3p_4, G, G_T, e)$, where $p_1p_2p_3p_4$ are distinct primes, G and G_T are cyclic groups of order N, And $e: G \times G \rightarrow G_T$ is a bilinear map such that:

   - (Bilinear) $\forall g, h \in G$ and $x, y \in Z_N$, it satisfies $e(g^x, h^y) = e(g, h)^{xy}$.
   - (Non-degenerate) $\exists g \in G$ such that $e(g, g)$ has order $N \in G_T$.

We require that the group operations in G and G_T and the bilinear map e are all computable in polynomial time. Let $\mathrm{Gp}_1, \mathrm{Gp}_2, \mathrm{Gp}_3$ and Gp_4 denote the subgroups of G with orders p_1, p_2, p_3 and p_4, respectively. Note that if $g_i \in Gp_i$ and $g_j \in Gp_j$ for $i = j$, then $e(g_i, g_j) = 1$. If the generator of Gp_j is $g_i (i \in \{1,2,3,4\})$, then every element $h \in G$ can be expressed as $g_1^{a_1} g_2^{a_2} g_3^{a_3} g_4^{a_4}$ for some values $a_1, a_2, a_3, a_4 \in Z_N$.

2) Access Tree: Let T be a tree representing an access structure. Each non-leaf node of the tree represents a threshold operator, which is described by its children and a threshold value If num_x is the number of children of node x, and k_x is its threshold value, then $1 \leqslant k_x \leqslant num_x$. When $k_x = 1$, the threshold is an OR operator, and when $k_x = num_x$, it is an AND operator. Each leaf node x of the tree is described by an attribute and a threshold value $k_x = 1$ [2]. Let T be an access tree with root r. The subtree of T rooted at node x is denoted by T_x. Thus, T is the same as T_r. If a set of attributes ω satisfies the access tree T_x, we denote it as $T_x(\omega) = 1$. We compute $T_x(\omega)$ recursively as follows: If x is a non-leaf node, we evaluate $T_x(\omega)$ for each child x of node x. $T_x(\omega)$ returns 1 if and only if at least k_x children return 1. If x is a leaf node, then $T_x(\omega)$ returns 1 if and only if $att(x) \in \omega$, where $att(x)$ denotes the attribute associated with node x [2].

B) CP-ABE Algorithms

CP-ABE consists of the following algorithms [2]:

- Setup (U). This algorithm takes as input an attribute universe U. It will initialize the system and generate the master key MK and the public key PK.
- KeyGen (PK, MK, ω). This algorithm takes as input the public key PK, the master key MK and a users attribute set ω. It will output a private key SK_ω.
- Encryption (PK, M, T). This algorithm takes as input the public key PK, a message M and an access-policy tree T. It will produce a ciphertext CT.
- Decryption (SK_ω, CT). The decryption algorithm takes as input a private key SK_ω and a Ciphertext CT. It will output the plaintext M if ω satisfies T.

4. SYSTEM MODEL AND THREAT MODEL

A. SYSTEM MODEL

We consider a file sharing system consisting of five parties: Trusted Authority (TA), Data Owner (DO), Data User, Fogs and the Cloud. In this system, the TA is responsible for

system initialization, authenticating the users´s attributes, creating and sending the secret keys to the users and the generating intermediaries keys to the fogs. The Data owner is the user who wants to upload and share his data; it is also his role to specify the access policy which is used to encrypt the data.The policy is used to controle who can access to this shared data. The data user is the one who wants to access to the shared data; he solicits the TA by sending his attributes in order to obtain a private key that will be used to decrypt the data. The cloud provides a storage service to users so that the shared data can be accessed anywhere and anytime. Fogs are entities to that collaborate and help users to partially decrypt the data. The system model is shown in Figure.1.

Our proposed model secure data sharing system by using following functions.

1. Setup$(\tau, \mathrm{N}, \mathrm{f}) \rightarrow \{\mathrm{PK}, \mathrm{MSK}\}$: the TA executes the setup function which takes as input a security parameter τ, the set of universal attributes N and the number of available Fogs f. As a result, it outputs a public key (PK) and a master secret key (MSK). The public key PK is known by all entities of the system.

2. Keygen$(PK, MSK, S, F) \rightarrow \{SK, TK\}$: this function is executed by the TA. When the user requests his private key by sending his set of attributes S, the TA generates two keys, a secret key (SK) and an intermediate key (TK). The latter operation is used after checking the validity of the attributes. The SK key is sent to the user while the TK key is sent to the fogs (F) and is used in the partial decryption phase. Sending these keys is performed through secure channels.

3. Encryption $(PK, M, T, L) \rightarrow \{E, CT_i\}$: the user encrypts the message M by specifying an access policy in tree form (T) and outputs a ciphertext (E) , in addition to several sub-tree CT_i according to the available fogs list L.

4. DecrypPartial$(CT_i, TK_i) \rightarrow \{C_i, P_i\}$: using its intermediate key TK, the fog partially decrypts the ciphertext .$\{C_i, P_i\}$ are sent to the user.

5. Decryption$(C_i, P_i, SK, E) \rightarrow M$: the user decrypts the ciphertext CT using .$\{C_i, P_i\}$ wich are sent by all the fogs. By using his private key, the user can recover the message M.

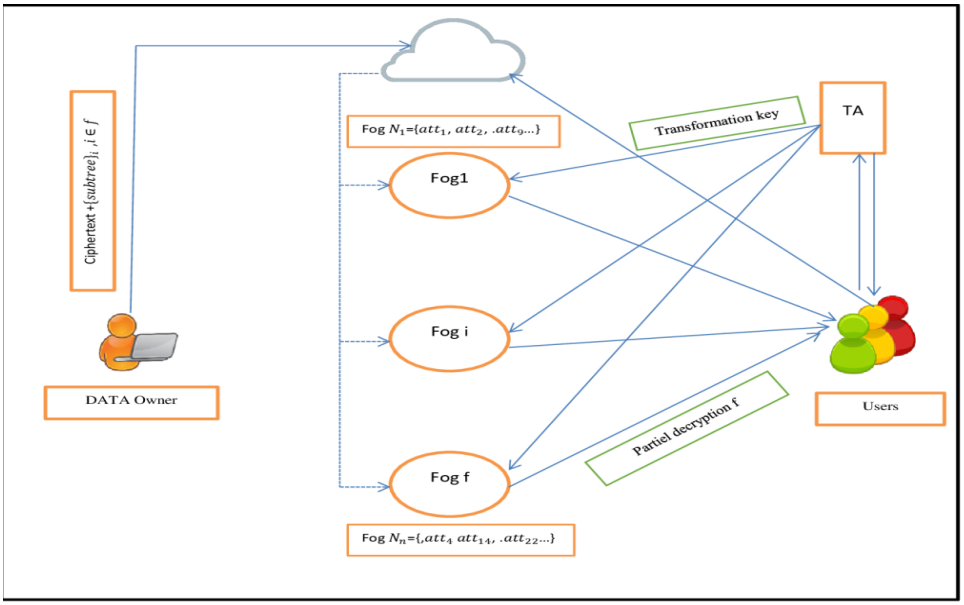

Figure 1. scheme of the proposed solution

B. THREAT MODEL

In our proposed system, we assume that TA is a trusted entity as in any system that uses ABE. We also assume that cloud and fogs are semi-trusted entities ie: the cloud and the fogs apply the protocols but curious entities.Also, we supposes that each fog manages a set of attributes in such a way that: $\forall i \neq jN_i \cap N_j = \emptyset$ where N_i, is the set of attributes belonging to the fog_i. The communication between the entities is secure.

5. DESCRIPTION OF THE PROPOSED APPROACH

In this section, we give a description of the diffrenet phases of our approach.

(A) Initialization phase : in this phase, the trusted authority generates two keys, a public key (PK) that is shared for all entities in the system and a master key (MSK) that will be kept secretly. After creating the keys, the TA assigns each user its own attributes. At the end of this step, each user will know the public key and the sets of all the attributes in the system.

(B) Encryption phase: when the Data Owner (DO) wants to share information with another user in the system according to an access policy, he creates an access policy T in tree form. He divides This tree into several subtrees T_i according to the available fogs and by taking into account the attributes managed by each fogs. The list of available fogs and their attributes is sent by the TA (Figure. 2). After obtaining the subtree T_i, the DO adds false attributes to hide the real attributes. He chooses random numbers $\{s_1 \dots s_f\}$ corresponding to each fog $\{Fog_1 \dots Fog_f\}$, where si is shared by all the attributes in T_i. Each s_i is shared for each node of the access tree T_i. The secret s_i is divided according to the "Top-Down approach" wheere the secret s_i is divided by $(t - n)$ Shamir secret sharing approach from the root to the leaf node. Where the parameter n is number of all child nodes and t is number of child nodes for recover secret s_i. Each real attribute in T_i will contain the shared λ_i of s_i. In contrast, the false attributes will not contain the share λ_i of s_i, moreover,

the false attributes will be eliminated in the partial decryption phase. After that, the DO sends the ciphertext with all T_i to the cloud for storage.

(C) Decryption phase: this phase contains two phases: the partial decryption and the final decryption. In the partial decryption, When a user wants to access to the shared data, he requests his private key from TA with his attributes (S). The TA chooses two random variable θ and t, where $(SK = \theta)$ will be the private key of the user and t it used with the set of users attributes to create the transformation keys TK_i for each available fog_i . The fogs can use TK to decrypt the data partially. Both keys are sent securely. The partial decryption at the level of fog_i is performed with the TK_i key. Each fog_i decrypts the data partially without knowing which attribute participated in the partial decryption. Each fog sends partially decrypted data to the user to recover the message M. Finally in the final phase and After the user receives all partially decrypted data, he recovers the message with his private key SK.

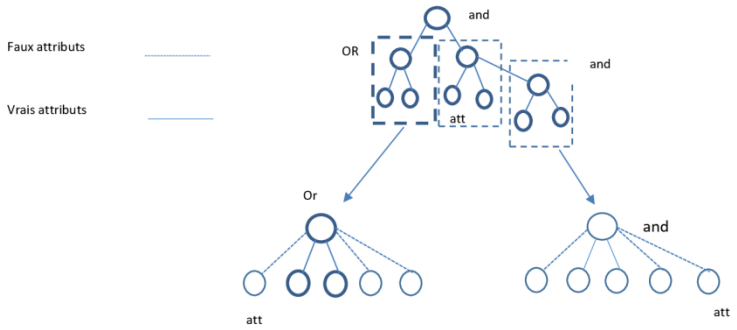

Figure 2. Division of access policy into several subtrees.

6. CONSTRUCTION

The algorithm starts by the setup phase until decryption phase. There are six phases .Each phase is detailed in the following paragraphs:

Initialization phase

(A) **Setup (τ, N, f)** : the algorithm takes as input a security parameter τ, the set of universal attributes N and the number of available Fogs f. The algorithm chooses a bilinear group G with an order $O = p_1p_2p_3p_4$, for each attribute $A_i \in N(1 \leqslant i \leqslant n)$ where n is the number of attributes in the universe N. Then is selects $h_i \in Z_N{}^*$ and finally selects a random element $\alpha, \beta \in Z_N^*$ and $g \in G_{p_1}$. The public key is defined by:

$$PK = \{N, g, y = (g,g)^{\alpha}, L = g^{\beta}, H_i = g^{h_i}(1 \leq i \leq n)\}$$

and the master key by:

$$MK = \{\alpha, \beta\}.$$

(B) The algorithm divides the set N by the number of available fogs f. This means $N = N_1 \cup N_2 \cup, \ldots, \cup N_f$ in such a way $\forall i \neq j, N_i \cap N_j = \emptyset$ where N_i is the set of attributes belonging to the fog_i This phase is executed by the trusted authority (noted TA in **Figure. 1**).

(C) When the user requests his private key with his set of attributes. The TA chooses a random variable θ that will be the private key of the user($SK = \theta$).

Encryption phase

(A) In this phase, the DO executes the Encryption primitive denoted **Encryption(PK, M, T, L)** as follows: The Encryption algorithm takes as input the public key PK , the message M and the access policy T in the tree form and L which represents the list of available Fogs with their attributes(N_i).

(B) The algorithm divides the tree T into several subtrees T_i according to the number of available Fogs. Each subtree will contain the attributes of the destination Fog.

(C) The Sender adds false attributes(nodes) to the subtrees according to the universe of attributes of the destination Fog. Let U_i be the set of attributes of the subtree T_i after adding false attributes. In the subsequent step, The Sender chooses a random numbers $\{s_1 \dots s_f\}$ corresponding to each fog $\{Fog_1 \dots Fog_f\}$ where s_i is shared by all the attributes in T_i.

(D) The algorithm shares the secret s_i as follows: a polynomial $q_i(x)$ degree $k_i - 1$ is chosen for each node (including the leaf node) in T_i where $k_i = |T_i|$ (number of elements in(T_i).These polynomials are generated in a recursive manner starting from the root node r. We define $q_{ir}(0) = s_i$(where r represent the root node in the tree) then other value $k_i - 1$ are defined randomly to complete the construction. Once all the polynomials have been defined we put $\lambda_{xi} = q_{xi}(0)$ for each node x in T_i, we choose random elements $Z_0, \{Z_i\}_{A_i \in U_i} \in G_{p4}$ knowing that $att(x) = A_i$ and $index(y)$ is attributes index of y in T_i , the ciphertext is generated as follows:

$$CT = \{E = My^s, E_0 = g^s Z_0, CT_i\}$$

$$CT_i = \begin{Bmatrix} \forall A_i \in T_i : E_i = L^{\lambda_{xi}} H_i^{s_i} Z_i \\ \\ \forall A_i \notin T_i : E_i = H_i^{s_i} Z_i \end{Bmatrix}, T_i$$

Where $s = \sum s_i$. The ciphertext is formed as CT include CT_i that is stored in the cloud.

Decryption phase

(A) When a user wants to access the shared data, he sends a request to the cloud about the encrypted data and request the TA to create the transformation keys TK. So, the TA executes the primitive **KeyGen(PK, MK, S, θ, f)** as follows: the TA (Trusted autorithy) creates the transformation keys TK for each fog. For that, the TA starts the key generation procedure where this key makes it possible to perform a partial decryption. To create TK **KeyGen** chooses a random element $t \in Z_N^*$ and $R, R_0, \{R_i\}_{A_i \in S_i} \in G_{p3}$, then returns the transformation key for each Fog. Formally:

$$TK = \{D = g^{(\alpha - \beta t)\theta} R, D_0 = g^t R_0, \forall A_i \in S_i : D_i = H_i^t R_i\}$$

Finally, the TA distributes the TK_i key to fog_i.

(B) Upon receipt of the TK_i key and CT_i , fog_i executes the following function:

DecrypPartila(CT_i, TK_i) : This algorithm takes as input CT_i and TK_i. When the fog_i receives CT_i it uses its transformation key TK_i to partially decrypt the ciphertext. Two recursive functions are used:

DecryptNode_$CT_i(CT_i, x)$ which takes as input CT_i and the node x which belongs to T_i.
DecryptNode_$TK_i(TK_i, x)$ which takes as input the transformation key and the x node.

The algorithm of **DecryptNode_CT_i** and **DecryptNode_TK_i** is defined by the following instructions:

If the node x is a leaf node Set $DecryptNode\_CT_i(CT_i, x) =$

$$E_i = \begin{cases} Ł^{\lambda_{xi}} H_i^{s_i} Z_i & if A_i \in T_i \\ H_i{}^{s_i} Z_i & if A_i \notin T_i \end{cases}$$

$$DecryptNode\_TK_i(TK_i, x) = D_i = H_i^t R_i$$

We consider the case where x is an internal node. The two functions $DecryptNode\_CT_i$, and $DecryptNode\_TK_i$ are executed in the following way: (knowing that the direction of execution is root to down) For each node y that is the child of x $DecryptNode\_CT_i$ and $DecryptNode\_TK_i$ are invoked. The result is saved respectively in F_y and K_y, let Q_x a set of y nodes child that belongs to T_i and $Q_{x'}$ the set of y nodes that does not belong to T_i. We have $Q_x \cup Q_{x'} =$ all the children of the x in the T_i tree. If y is a node then we calculate:

$$\begin{aligned} F_x &= \prod_{y \in Q_x \cup Q`_x} F^{l_y v_x(0)} \\ &= \prod_{y \in Q_x} \left(L^{\lambda_{xi}} H_i^{s_i} Z_i\right)^{l_y v_x(0)} \cdot \prod_{y \in Q_x \cup Q`_x} \left(H_i^{s_i} Z_i\right)^{l_y v_x(0)} \\ &= \prod_{y \in Q_x} g^{\beta \lambda_{yi} . l_y v_x(0)} . \prod_{y \in Q_x \cup Q`_x} H_i^{si . l_y v_x(0)} . \prod_{y \in Q_x \cup Q`_x} Z_i^{l_y v_x(0)} \\ &= g^{\beta \lambda_{xi}} . F_{x,1} . F_{x,2} \end{aligned}$$

And

$$\begin{aligned} K_x &= \prod_{y \in Q_x \cup Q`_x} . K^{l_y v_x(0)} \\ &= \prod_{y \in Q_x \cup Q`_x} . H_i^{t . l_y v(0)} \cdot \prod_{y \in Q_x \cup Q`_x} . R_i^{l_y v_x(0)} \\ &= K_{x,1} . K_{x,2} \end{aligned}$$

If the node is non-leaf node we calculate:

$$\begin{aligned} F_x &= \prod_{y \in (Q_x \cup Q`_x)} . \left(g^{\beta \lambda_{yi}} . F_{x,1} . F_{x,2}\right)^{l_y v_x(0)} \\ &= \prod_{y \in Q_x} (g)^{\beta \lambda_{yi} . l_y v_x(0)} . \prod_{y \in Q_x \cup Q`_x} . F_{y,1}^{l_y v_x(0)} \cdot \prod_{y \in Q_x \cup Q`_x} . F_{y,2}^{l_y v_x(0)} \\ &= g^{k \lambda_{xi}} . F_{x,1} . F_{x,2} \end{aligned}$$

And

$$\begin{aligned} K_x &= \prod_{y \in Q_x \cup Q`_x} . K^{l_y v_x(0)} \\ &= \prod_{y \in Q_x \cup Q`_x} . K_{y,1}^{l_y v_x(0)} \cdot \prod_{y \in Q_x \cup Q`_x} . K_{y,2}^{l_y v_x(0)} \\ &= K_{x,1} . K_{x,2} \end{aligned}$$

In the previous equation, we have $F_{x,1} = K_{x,1}$ the parameter $v_x = \{index(y)/y \in (Q_x \cup Q`_x)\}$ and $l_y v_x(0)$ is the coefficient of lagrange. If we call both functions from root r of T_i then we obtain:

$A = \text{DecryptNode\_}CT_i(CT_i, r) = g^{ks_i}.F_{r,1}.F_{r,2}$
And
$B = \text{DecryptNode\_}TK_i(TK_i, r) = K_{r,1}.K_{r,2}$

We calculate:

$$C_i = e(A, D_0)/e(E_0, B)$$
$$= e(g^{\beta s_i}.F_{r,1}.F_{r,2}, g^t R_0)/e(g^{s_i}Z_0, K_{r,1}.K_{r,2})$$
$$= e\left(g^{\beta s_i}, g^t\right).e\left(F_{r,1}, g^t\right).e(F_{r,2}, g^t).$$
$$e(g^{\beta s_i}.F_{r,1}.F_{r,2}, R_0)/e(g^{s_i}, K_{r,1}).e(g^{s_i}, K_{r,2}).e(Z_0, K_{r,1}.K_{r,2})$$
$$C_i = e(g, g)^{\beta t s_i}.$$

And also:

$$P_i = e(E_0, D)$$
$$= e(g^{s_i}Z_0, g^{(\alpha-\beta t)\theta}R)$$
$$= e\left(g^{s_i}, g^{(\alpha-\beta t)\theta}R\right)$$
$$= e(g, g)^{s_i(\alpha-\beta t)\theta}$$

(C) Finally, the fog sends the partial decryption C_i and P_i to the user.
(D) Upon receipt of all shares parts of fog_i, the user executes the following function:

Decryption(C_i, P_i, SK, E) : this algorithm is executed by the user. If the user receives all the parts which are partially decrypted from the Fog, then he knows that his attributes satisfy the access policy

Otherwise, he rejects the decryption. When the user receives all the parts which was partially decrypted, he uses his private key Sk= θ and the ciphertext transformed by the Fog (C_i, P_i) to recover the original message.

Formally:

$$\frac{E}{(\prod P_i)^{\frac{1}{\theta}}.\prod C_i} = \frac{E}{(\prod e(g,g)^{s_i(\alpha-\beta t)\theta})^{\frac{1}{\theta}}.\prod e(g,g)^{\beta t s_i}}$$

$$= \frac{E}{(e(g,g)^{(\alpha-\beta t)\theta\sum s_i})^{\frac{1}{\theta}}.e(g,g)^{\beta t\sum s_i}}$$

$$\frac{E}{(e(g,g)^{s(\alpha-\beta t)\theta})^{\frac{1}{\theta}}.e(g,g)^{s\beta}} = \frac{E}{e(g,g)^{s(\alpha-\beta t)}.e(g,g)^{st\beta}} = \frac{E}{e(g,g)^{s\alpha}} = \frac{Me(g,g)^{s\alpha}}{e(g,g)^{s\alpha}} = M$$

7. ANALYSES

In this section, we discuss the security properties of the proposed solution involving data privacy, fine-grained access control, and collision resistance.

(A) Security Proprieties

(1) **Data confidentiality**: The confidentiality requires that the cloud and the fog cannot learn the encrypted data, in the decryption-outsourcing algorithm; the cloud is responsible only for storing the encrypted data as $M.e(g,g)^s$ where s is kept secret by the user. While, the Fogs are

responsible only for the partial decryption of the data, and since the transformation keys TK_i are generated by TA with the secret key of the user, only the end user where his attributes correspond to the access policy, can recover the encrypted data, in other words, fogs cannot recover random value s where this value is divided among the fogs in the encryption process even if the fogs cooperate with each other, since in the processing of the partial decryption the S_i are blinded with the secret key of the user $SK = \theta$. Thus, we conclude that our scheme is secure in protecting the confidentiality of the message.

(2) **Hidden access policy**: in our schema the DO adds false attributes to the access policy, with this method the malicious users and even the Fogs cannot have the real attributes even if the Fogs cooperate with each other, this will lead to the addition of several false attributes which further complicates the task of having the right attributes. Also, the Fogs and even the users cannot know which attribute participated in the decryption of the data as all the attributes of the users whether they belong to the access policy are being applied in the decryption process.

(3) **Fine-grained access control**: the proposed solution uses the CP-ABE algorithm, where the DO defines an access policy for each outsourced data. This access policy is in the form of an And-gate tree where the tree contains the attributes that allow access to the data, in this way only the users that their attributes match with the attributes in the access policy can decrypt the data.

(4) **Collusion resistance:** the collision resistance is the property that the CP-ABE assumes. In our solution, the algorithm Keygen generates a different random values t for each user and which is integrated into the key of transformation. It means that each key of the user is randomized; this means that users cannot combine their keys to decrypt the data, so malicious users cannot collaborate to expand their access privileges including fog nodes since the transformation key contain the random value t .

(A) Analysis and discussion:

An overview comparison of some existing CP-ABE schemes with our scheme is presented in Table1 .Our scheme achieves policy hiding ,fast decryption, outsourced decryption process and proven fully secure in the standard model . The access policy is specified based on the tree structure which allows the data owner to specify a complex access policy in intuitionistic form, There by delivering a better user experience than LSSS. The comparison indicates that our scheme has all of the following features: hidden policy, fast decryption, outsourced decryption, expressivity and full security.

Table1. an overview comparison

Scheme	Access Structure	Policy hidden	Fast decryption	outsourced computation	security
[4]	Tree	No	No	Yes	Selective
[5]	Tree	No	No	Yes	Selective
[6]	Tree	No	No	Yes	Selective
[7]	LSSS	No	No	Yes	Selective
[8]	Tree	Yes	Yes	No	Full
[9]	LSSS	Yes	No	Yes	Selective
Our approach	Tree	Yes	Yes	Yes	Full

(B) Performance Analysis

In this section, we compare our scheme against two approaches: (1) traditional CP-ABE and (2) the scheme proposed by Wang and Lang [8]. Our comparative study, illustrated in Table 2, is based on the decryption complexity at the user level. This choice is motivated by the fact that we used partial decryption that is delegated to Fogs which have unlimited recourse in terms of energy and computing capacity.

Table2. computation cost

Scheme	Complexity of decryption
CP-ABE	(2n+1)P+2M
[8]	3P+2(n)E
Our schema	1 E+2(F)M

Modular exponentiation (E) and bilinear pairing (P) are two computationally expensive operations in attributes based encryption. We utilize the number of E and P as measurements to evaluate the performance of our scheme According to Table 1, we see that the decryption cost in the traditional CP-ABE scheme is significant. The user executes (2n + 1) pairing operation (P), where n the number attributes in the access policy. In addition, the user also executes 2M Where M denotes the multiplication group in G. Unlike the approach in [8] where the user executes 3P and 2(n)E . However, in our scheme, the user executes only 1E exponentiation and 2(F)M and the fog execute 3p + 2(n)E where n denote the number of attributes in CTi and F denotes the number of available Fogs in the scheme. Knowing that multiplication consumes less than paring and exponentiation operations we notice that our scheme improved of decryption at the user level compared to other scheme mentioned above. However, the increase of computation on fog side which should be insignificant for the fog.

8. APPLICATION SCENARIOS

In this section, we introduce an application scenario. Our schema can be used in healthcare systems, where wearable devices can detect and collect users health data. The system is composed of entities such as medical insurance, analysis laboratory, private hospitals, hospitals, where each entity manages a set of attributes. In addition, each entity is connected to a fog that will manage these attributes. A doctor or a member of the patient's family is authorized to decrypt the data (according to the access policy). Because the doctor or the family member use constrained devices, they request the fog to partial decrypt the data. According to our proposed method, the scenario is defined as follows:

1) After receiving the collected data on his smartphone, the patient (or data owner) defines -by utilizing an application GUI for example- the access policy which specifies who can access the data. for Example a doctor .

2) Then the device splits the access policy by taking into consideration attributes that are managed by each fog. Next, it pads them with false attributes and sends each part to the corresponding fog.

3) The data file is encrypted and sent to the cloud along with the complete access policy with is also padded with false attributes.

4) When a doctor wants to read the data file, he connects to the cloud to get this file. His attributes are sent to the trusted authority which will create the intermediate key.

5) This intermediate key is sent to the fog nodes which partially decrypt the ciphertext. This process also includes testing the partial access policy (see CP-ABE section)

6) After decryption, all fog nodes send the partial ciphertext to the doctor which ecrypts the complete ciphertext using his private key.

9. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a new collaborative approach based on CP-ABE. In our approach, we used the Fog nodes to reduce the bandwidth and to decrease the decryption cost by delegating the decryption process to the fog nodes. This allowed us to reduce the complexity (at the user level) to one exponentiation and multiplications operations instead of the paring operations which are energy-intensive.

Our solution also preserves the privacy of the access policy so that the data owner attribute information is not disclosed. This is performed by introducing false attributes which are mixed with the real attributes.

As future work, we plan to evaluate our approach on real devices. Our work also improved by using more expressive access policies (i.e., policies with ANDs and ORs). Bandwidth can also be optimized by reducing the number of communications between the Fogs and user.

REFERENCES

[1] Hany F. Atlam, Robert J. Walters, and Gary B. Wills. Fog computing and the internet of things: A review. Big Data and Cognitive Computing, 2(2), 2018.

[2] J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In 2007 IEEE Symposium on Security and Privacy (SP '07),pages 321–334, Los Alamitos, CA, USA, may 2007. IEEE Computer Society.

[3] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attributebased encryption for fine-grained access control of encrypted data. In Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS '06, pages 89–98, New York, NY, USA, 2006.ACM.

[4] Zhibin Zhou and Dijiang Huang. Efficient and secure data storage operationsfor mobile cloud computing. In Proceedings of the 8th International Conference on Network and Service Management, CNSM '12, pages 37–45, Laxenburg, Austria, Austria, 2013. International Federation for Information Processing.

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

[6] Kim Thuat Nguyen, Nouha Oualha, and Maryline Laurent. Securely outsourcing the ciphertext-policy attribute-based encryption. World Wide Web, 21(1):169–183, January 2018.

[7] Kai Fan, Junxiong Wang, Xin Wang, Hui Li, and Yintang Yang. A secure and verifiable outsourced access control scheme in fog-cloud computing. Sensors, 17(7), 2017.

[8] Jinmiao Wang and Bo Lang. An efficient and privacy preserving cp-abe scheme for internet-based collaboration. In CollaborateCom, 2017.

[9] Sana Belguith, Nesrine Kaaniche, Maryline Laurent, Abderrazak Jemai, and Rabah Attia. Phoabe: Securely outsourcing multi-authority attribute based encryption with policy hidden for cloud assisted iot. Computer Networks, 133:141 – 156, 2018.