future internet

MDPI

Article

Towards Virtuous Cloud Data Storage Using Access Policy Hiding in Ciphertext Policy Attribute-Based Encryption

Siti Dhalila Mohd Satar [1,2], Masnida Hussin [1,*], Zurina Mohd Hanapi [1] and Mohamad Afendee Mohamed [2]

[1] Department of Communication Technology and Networks, Universiti Putra Malaysia (UPM), Serdang 43400, Selangor, Malaysia; sitidhalila@unisza.edu.my (S.D.M.S.); zurinamh@upm.edu.my (Z.M.H.)
[2] Centre of Computer Science Study, Universiti Sultan Zainal Abidin, Kampus Besut, Besut 22200, Terengganu, Malaysia; mafendee@unisza.edu.my
* Correspondence: masnida@upm.edu.my

Abstract: Managing and controlling access to the tremendous data in Cloud storage is very challenging. Due to various entities engaged in the Cloud environment, there is a high possibility of data tampering. Cloud encryption is being employed to control data access while securing Cloud data. The encrypted data are sent to Cloud storage with an access policy defined by the data owner. Only authorized users can decrypt the encrypted data. However, the access policy of the encrypted data is in readable form, which results in privacy leakage. To address this issue, we proposed a reinforcement hiding in access policy over Cloud storage by enhancing the Ciphertext Policy Attribute-based Encryption (CP-ABE) algorithm. Besides the encryption process, the reinforced CP-ABE used logical connective operations to hide the attribute value of data in the access policy. These attributes were converted into scrambled data along with a ciphertext form that provides a better unreadability feature. It means that a two-level concealed tactic is employed to secure data from any unauthorized access during a data transaction. Experimental results revealed that our reinforced CP-ABE had a low computational overhead and consumed low storage costs. Furthermore, a case study on security analysis shows that our approach is secure against a passive attack such as traffic analysis.

Keywords: CP-ABE; fine-grained access control; policy hiding; privacy-preserving

Citation: Mohd Satar, S.D.; Hussin, M.; Hanapi, Z.M.; Mohamed, M.A. Towards Virtuous Cloud Data Storage Using Access Policy Hiding in Ciphertext Policy Attribute-Based Encryption. *Future Internet* **2021**, *13*, 279. https://doi.org/10.3390/fi13110279

Academic Editor: Rattikorn Hewett

Received: 15 September 2021
Accepted: 16 October 2021
Published: 30 October 2021

1. Introduction

Cloud computing has become a priority and integral to modernizing the information technology (IT) environment. It spawned a whole new dimension of IT, which utilized a wide range of resources that contributed to many domains, such as business, military, health, and medical. In the healthcare and medical domain, Cloud computing has been adopted to facilitate day-to-day operations. This is because the Cloud provides 'on-the-fly' services, that is, storage, computation, and data sharing, which allow healthcare and medical practitioners to run their business according to their required operations. For example, electronic health records (EHRs) have been widely employed in the healthcare industry to improve the accessibility and sharing of medical data among medical practitioners. Patients' information, laboratory results, medication lists, diagnostic tests, physical examinations, and historical observations are all kept in the EHR, which is stored in Cloud storage. Furthermore, resource management and system administration (infrastructure) can be effectively monitored through Cloud computing, making healthcare services easy to maintain [1]. Besides, Cloud Service Providers (CSPs) facilitated Cloud users by granting resource sharing regardless of geographical boundaries using the pay-per-use model [2–4], which can reduce organizations' management and maintenance costs. Moreover, Cloud adoption in organizations is expected to lift performance with the high-speed deployment of services and to improve clients' satisfaction.

Organizations are willing to adapt their business operations to the Cloud environment due to ultimately easier access to Cloud services and numerous benefits. One of the areas

that organizations particularly choose is using Cloud storage to afford them the ability to access their files (data and information) from any device and any place. Another reason is by making Cloud storage to be part of the backup solution. In addition, assessing and sharing data from the Cloud environment offers highly available services. That feature is crucial for organizations dealing with numerous clients besides diverse roles and demands. Once Cloud computing is established with its core services (i.e., IaaS, PaaS, SaaS), Cloud users begin to be concerned about security matters [5–8]. From the Cloud storage perspective, data confidentiality and privacy have become debatable among Cloud users due to the skeptical location where the files reside. Furthermore, when they noticed that the CSPs operated in a multi-tenant environment, the organizations raised the demand for security features from the CPSs. It is mainly to ensure the users' data and information stored in the Cloud are not exposed or disclosed to any illegitimate access.

Since early in the year of 2000, many types of research have been developed in tackling the Cloud security issue. Among security approaches, encryption has initially become the essence of the Cloud security call. The authors of [9–11] proposed the encryption algorithm for securing Cloud data that converts the data into unreadable forms. Therefore, any attempt to sniff data over cloud storage can be prevented. However, studies by the authors of [12,13] stated that the encryption algorithm alone does not guarantee data safety in Cloud storage. In fact, in a multi-tenant environment, the probability of data exposure is high because different users share the same Cloud infrastructure and storage. It means that they stored the data in the same location, and more crucially, it was probably kept in the same stack of the server. This leads to unauthorized access by other users [14,15], especially when those users have the intention of hacking and stealing the data. Thus, besides performing data encryption, access control over Cloud storage needs to be addressed.

Therefore, other researchers began to bring together encryption with other solutions, such as privacy-preserving schemes and access control schemes [16], to enhance data privacy further. Attribute-based Encryption (ABE) is a promising solution that offers fine-grained access control and provides data confidentiality on Cloud storage [17]. Sahai and Waters [18] were the pioneers of Attribute-based Encryption (ABE), where they claimed that the fine-grained access control scheme was able to support better security services for Cloud users and CSPs. In ABE, different attributes of various Cloud environment entities, for example, data owner, file, and data recipient, are used to describe the encrypted data and built policies into the user's keys. Later, the authors of [19] introduced Ciphertext Policy ABE (CP-ABE), which uses attributes to describe a user's credentials, and a party encrypting data determines a policy for who can decrypt the ciphertext. Basically, in CP-ABE, the access policy created for the encrypted message is sent in a plaintext format. It provides the opportunity for illegal parties to retrieve the attribute details in the access policy, subsequently disclosing the data. Besides, several other issues related to CP-ABE have also been discussed in [20]. Thus, many researchers [6,19,21,22] introduced new CP-ABE schemes to solve the issues. The authors of [18] introduced a CP-ABE scheme to enhance the efficiency of data sharing between data owners and other users. It also allowed data owners to define the access policy of the encrypted data so that only users who match the access policy can download and reveal the data. According to [6], most proposed CP-ABE schemes have accomplished their objectives by providing efficient and secure data sharing. However, most of these constructions disregard the privacy of data owners and data users. For instance, in the healthcare system (Figure 1), a doctor needs to share his patient's health record with other doctors. Based on the conventional method of CP-ABE, the nurses could also have access to cloud storage to obtain patient information from the attributes in the access policy. It is due to the access policy that is partially hidden and can be read by authorized entities, which might turn into malicious access. Hence, it is crucial to enforce the encryption scheme to not merely encrypt the message but entirely hide the access policy details.

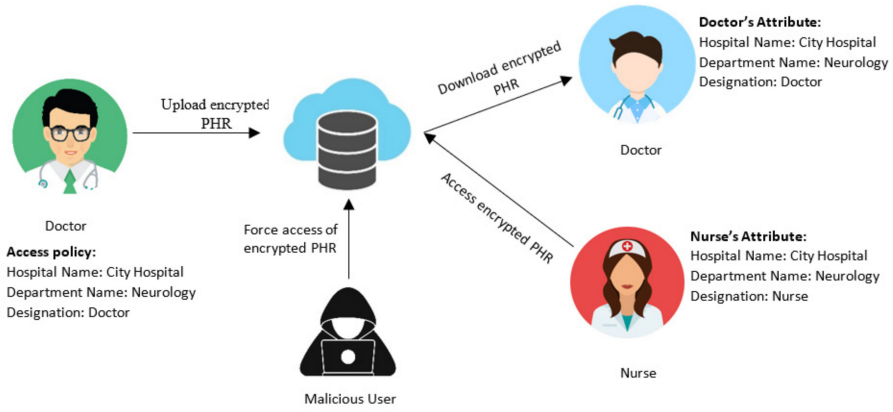

Figure 1. Sample Privacy Leakage State.

Therefore, inspired by the literature [23], we present a Policy Hiding using Logical Connective in CP-ABE (PHLC) scheme for Cloud storage, which adopts XOR operation to modify access policy information. This proposed work can overcome the encryption challenges outlined by [24] since it was designed to provide data confidentiality using a symmetric encryption scheme and offered fine-grained access control with data sharing provision. Besides, this scheme preserved users' privacy through an access policy hiding scheme. In addition, PHLC achieved efficient data storage utilization by using a pre-processing process to eliminate redundant data in raw shared data. This operation helps the scheme to reduce ciphertext size and decrease the computational overhead for the encryption process. Security analysis shows that the proposed scheme preserved Cloud users' privacy and guaranteed secure data sharing in cloud storage. Furthermore, we conducted an extensive simulation to demonstrate that PHLC CP-ABE was secure against passive attacks.

The rest of the paper is arranged as follows. Section 2 discusses related work and Section 3 provides some preliminaries of CP-ABE, whereas Section 4 presents the proposed scheme's implementation. Section 5 discusses security analysis and Section 6 provides results and discussion. Finally, Section 7 concludes this study.

2. Related Work

Numerous works of research such as revocable CP-ABE [25], lightweight CP-ABE [26], multi-authority CP-ABE [4,27], and large universe CP-ABE [28] have been developed in response to upgrading the competency of the Ciphertext Policy Attribute-based Encryption (CP-ABE) scheme. However, most of the schemes exposed the access policy in plaintext, which incurred privacy leakage. As a result, there are other works that focus on encrypting the data while hiding the access policy during data sharing. Nishide et al. [29] proposed the hiding access policy by splitting attributes into attribute names and multiple attribute values, where they then hid the attribute values. Meanwhile, Zhou et al. [30] proposed the partially hidden access algorithm, whereby the hidden access policies are implemented with wildcards regardless of the number of attributes. Although the data construction effectively secures the shared information, it fails to offer total data security. It is due to the wildcard attributes not yet being in readable form and might cause privacy leakage. The authors of [23] supported fast decryption while hiding the access policy by sending the access matrix and the defined function along with ciphertext to the Cloud environment. However, it is unable to preserve privacy in the access policy. In [31], the authors eliminated all redundant attributes in the access policy, and their approach significantly reduced computation overhead. With the same intention, the author of [32] proposed the 'test-decrypt-verify' approach to reduce the computation cost in their CP-ABE scheme. In their scheme, the testing phase is added before the encryption phase, and the new component, Outsourcing Cloud Server, is adopted as an outsource agent to reduce the decryption calculation. However, the proposed scheme only employed a partially hidden access

policy. The authors of [33] also provided a partially hidden access policy by removing the attribute name from the access structure of the ciphertext. However, the probability of data exploitation by dishonest entities or unauthorized users is still high because only a part of the access policy is hidden.

Other researchers expressed access policies using Linear Secret Sharing Schemes (LSSS) in the CP-ABE scheme for better data access control. In Lai et al. [11], they proposed partial hiding access structures with LSSS that are able to accommodate any access structure. They proved that their scheme was suitable for outsourcing data with attribute values of the data that had been hidden. Other studies, such as [34–36] also constructed the LSSS-based access policy schemes with CP-ABE, which hid attribute values to secure the information. Besides the partially hidden access policy, Xiong's scheme [34] supported attribute revocation and verifiable outsourced decryption. However, the attribute values carry more intricate information in comparison to generic attribute names, which leads to high computing overhead during the decryption process. Meanwhile, the authors in [35] proposed the control access scheme to provide privacy protection via partial concealment of access policy. They have utilized CP-ABE in the intelligent healthcare system known as the privacy-aware-health access control system (PASH). PASH hides the attribute value of the access policy in the encrypted Smart Health Record and only specifies the attribute name, which destroys the system-user privacy. Therefore, the existing CP-ABE scheme needs improvement to prevent information leakage from access policies and ensure data sharing on the Cloud is secured. Although various approaches have been proposed in the existing literature on the policy hiding scheme, the study on privacy-preserving with a fully hidden access policy is still inadequate. It could not solve the privacy leakage issues completely. Thus, the privacy-preserving of data users cannot be guaranteed.

3. Design of CP-ABE

This section discussed the preliminary work involved in designing the CP-ABE. In addition, it also covers the overview of CP-ABE, PHLC scheme's system model, security goals, and security model.

3.1. Preliminaries Works

This section presents the bilinear map, Linear Secret Sharing Scheme (LSSS), XOR-based Logical Connective, and notation definition.

3.1.1. Definition of Notations

This section gives notation explanations used in this research, as shown in Table 1.

Table 1. Notation.

Notation	Explanations
$\mathbb{G}$, $\mathbb{G}_T$	Two cyclic multiplicative groups
λ	Security parameter
AU	Attribute Universe
Att_n	An attribute
$\mathbb{A} = (\boldsymbol{A}, \rho, \mathcal{T})$	Access structure
$\boldsymbol{A}$	$l \times n$
ρ	ρ maps each row A_i of the matrix $\boldsymbol{A}$ to an attribute
$\mathcal{T}$	attribute value involved $\mathbb{A}$
$S = (Bs, Js)$	User's Attribute
Bs	attribute name index
Js	attribute value set
PK	Public Parameters

Table 1. *Cont.*

Notation	Explanations
MSK	Master Key
SK	Secret Key
M	Message (data sharing by the data owner)
CT	Ciphertext
$Att(x, y)$	location of the Attribute in attribute value involved in $\mathbb{A}$

3.1.2. Bilinear Pairing

In our CP-ABE scheme, bilinear pairing is used to create a public key. The Attribute Authority generates this public key based on a composite order bilinear group with a distinct prime order N, which we adopted from [23]. The algorithm takes an input 1^{λ} where λ is a security parameter and produces a tuple ($\mathbb{G}$, $\mathbb{G}_T$, e, $p1, p2, p3, p4$) where $p1, p2, p3, p4$ are distinct primes. The order of cyclic group $\mathbb{G}$ and $\mathbb{G}_T$ is $N = p1p2p3p4$, and map e: $\mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$, with properties:

i. Bilinearity: for all $g, y \in \mathbb{G}$ and $d, w \in \mathbb{Z}_N$, where $e(g^d, y^w) = e(g, y)^{dw}$
ii. Non-degenerate: there exists $g \in \mathbb{G}$ such that $e(g,g)$ has order N in $\mathbb{G}$
iii. Computable: e can be computed efficiently.

3.1.3. Linear Secret Sharing Schemes (LSSS)

The LSSS is used to express access policy in access structure (A, ρ), where A is a policy matrix and ρ is a mapping of each row A_i of the matrix A to an attribute [37]. In this scheme, the presence of an attribute universe was denoted as AU, which has n categories of attributes.

Definition 1 (LSSS). *Let $AU = (Att_1, Att_2, Att_3, \ldots, Att_n)$. Each attribute Att_n contains two-part which are attribute name and attribute values. Possible values of the attribute value, $AVx = \{\xi_{x,1}, \xi_{x,2}, \xi_{x,3}, \ldots, \xi_{x,nx}\}$. Meanwhile, $\boldsymbol{A} = \frac{l \times n}{\mathbb{Z}p}$ refers to a share-generating matrix, while each row in A is a map to an attribute name index, and that mapping was denoted as ρ. LSSS consisting of the two following algorithms:*

i. *Secret share: The secret shared, $s \ni \mathbb{Z}p$ and the value λ_x is computed for each row A_x of $\boldsymbol{A}$, where $V = (s, y_2, y_3, \ldots, y_n) - \in_R \mathbb{Z}_p^n$ and $y_2, y_3, \ldots, y_n$ are chosen randomly from $\mathbb{Z}p$. Hence, the secret share value is given $\lambda_x = A_x \times$ v.*
ii. *Secret Construction: This algorithm takes in the secret share $\{\lambda_x\}$ and set P which contains the authorized attribute name index. Then it sets $I = \{x | \rho(x) \in P\} \subseteq \{1, 2, \ldots, l\}$ and computes the constant $\{\omega_x\}_{x \in I}$ such that $\sum_{x \in I} \omega_x A_x = (1, 0, 0, \ldots, 0)$. Then the secret s is reconstructed by $s = \sum_{x \in I} \omega_x \lambda_x$.*

Similar to [23], we construct the LSSS matrices over $\mathbb{Z}_N$. In our proposed scheme, we denoted the user's attribute as $S = (B_s, J_s)$, where $B_s, \subseteq \mathbb{Z}_N$ is the attribute name index, and $J_s = \{l_x, i\}_x \subseteq B_s$ is the attribute value set. Denote $\mathbb{A} = (\boldsymbol{A}, \rho, \mathcal{T})$ as access policy and $\mathcal{T}$ is the attribute value for each row of A where $\mathcal{T} = \left(t_{\rho(1)}, t_{\rho(2)}, t_{\rho(3)}, \ldots, t_{\rho(l)}\right)\left(t_{\rho(x)} \in AV_{\rho(x)}\right)$. S satisfies $\mathbb{A}$ means that there exits $I \subseteq \{1, 2, \ldots, l\}$ satisfying $(A, \rho), \{\rho(x) | x \in I\} \subseteq B_s$ and $l_{\rho(x)} = t_{\rho(x)} \forall x \in I$.

3.1.4. XOR-Based Logical Connective Policy Hiding

The proposed policy hiding algorithm uses an XOR-based logical connective to hide the access policy in the CP-ABE scheme. We utilized XOR to modify the entire access policy to form a reliable proposition. According to Rosen [38], logical connective is defined by the truth table, which declares that a fact could be either true or false, but not both. Rosen described proposition logic as: 'Let p and q be propositions. The exclusive-or of p and q, denoted by p XOR q, is the proposition that is true when exactly one of p and q is true

and is false otherwise. According to [39], XOR is a lightweight operation that does not incur much computational overhead. Therefore, it is very appropriate to be adopted in this scheme as a second layer of protection.

3.2. Overview of Ciphertext Policy Attribute-Based Encryption

In this work, we enhanced the functionality of the CP-ABE scheme to embrace the fully hidden access policy. Therefore, four significant modules of CP-ABE were involved in constructing our hiding reinforcement approach.

System Setup (1^{λ}): It takes security parameter, 1^{λ} as an input, and produces a public parameter key PK, also a master key MSK as the outputs. They are used as input for the key generation algorithm.

Key Generation (PK, MSK, S): Attribute Authority is the main component in CP-ABE that we utilized to execute the key generation algorithm. The Attribute Authority used inputs from the system setup (i.e., public parameters PK, master key MSK, and users' attributes S) for generating a secret key SK. Later, the secret key is employed by the data user to decipher the encrypted data.

Encryption (PK, $M, \mathbb{A}$): In this module, it captures the public parameters PK, a message M and an access structure $(\boldsymbol{A}, \boldsymbol{\rho}, \mathcal{T})$ as the inputs. The encryption algorithm then produced the ciphertext CT. The data owner sends out the ciphertext along with the hashed value in the Cloud environment.

Decryption (PK, CT, SK): The decryption module took the public parameters PK, a secret key SK associated with the attributes set (A, ρ), and a ciphertext CT as the input. All these three inputs are used to decrypt the ciphertext CT and produce message M.

3.3. System Model

Figure 2 shows that the system model consists of four entities, which are Data Owner (DO), Cloud Service Provider (CSP), Attribute Authority (AA), and Data User (DU). The DO are legal users that are responsible for encrypting and outsourcing the data in the Cloud. The data owner defined the access policy and performed a policy hiding process. While in the pre-processing process, the redundant data in the raw message is eliminated before it is encrypted in the encryption process. Next, the data owner uploads the ciphertext with the hidden value of the access policy to the Cloud storage. Hence, the DU (or recipients) who satisfy the access policy will be able to decrypt the data. In this work, Cloud Service Providers (CSPs) are assumed to be solitary entities that do not interest their users/clients. It means acting as a platform where various data and users can reach, passing the messages, and storing the files. The access policy is handled on the user's side (DO). The Attribute Authority (AA) is an accountable entity that works as a key generation center. The users' attributes will be authenticated by AA before granting access privileges to the authorized users to interact with the system. The AA might receive authentication privileges from the DO or any other Cloud security mechanism agreements.

3.4. Security Goals

Our reinforcement hiding in access policy means to conceal the attributes' details. Hence, it ought to consider the security features as follows:

Confidentiality: It is achieved when unauthorized users are unable to access the encrypted data, and only users who satisfy the access policy can perform the encryption and decryption module. Data confidentiality is also achieved when other entities, including the CSPs, cannot read/access any information from the encrypted data.

Data privacy: The access policy in our scheme complied with the encrypted data features where it had been hidden even though it was already in the ciphertext. It is performed in a two-level data concealment strategy that fully hides the attributes and data compression. When it reaches its destination, the decryption is performed with the respective key to disclose the data in a readable form. Therefore, it prevented the user's privacy from being exposed.

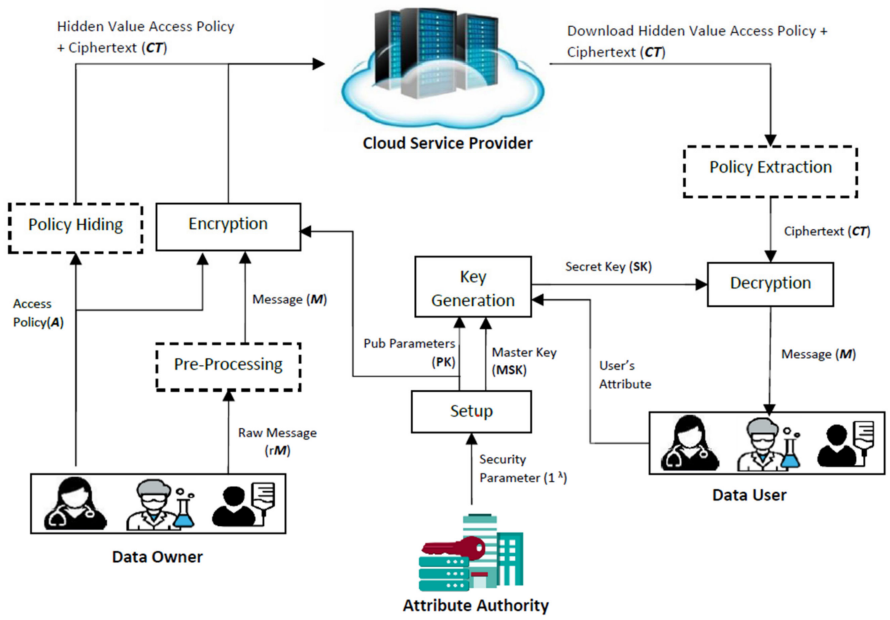

Figure 2. System Model.

Fine-grained access control: Users of the Cloud do not have the equal privilege of retrieving data. The privilege depends on the extent to which the user is involved or responsible. In our design, users are assigned to dissimilar access privileges defined based on the access policy imposed by the Attribute Authority (AA). All the attributes should be matched with the user access policy structure to retrieve the required information.

3.5. Security Model

This section discusses the security model for the proposed PHLC scheme. This scheme was constructed based on Zhang's scheme [23] by re-simulating their works. Hence, this scheme's security model is based on a security game between adversary $\mathcal{A}$ and challenger $\mathcal{B}$ as presented in [23]. The following is a full description of the game:

Setup. To obtain the public parameters PK and the master key MSK, Challenger $\mathcal{B}$ runs the setup algorithm. The Challenger $\mathcal{B}$ holds the master key MSK and sends adversary $\mathcal{A}$ the public parameters PK.

Phase 1: Adversary $\mathcal{A}$ adaptively issues a secret key query to the key generation module. For each query on an attribute set S_i, challenger $\mathcal{B}$ returns a Secret Key SK_{si} to Adversary $\mathcal{A}$.

Challenge. Adversary $\mathcal{A}$ generates two messages M_0^*, M_1^*, ($|M_0^*| = |M_1^*|$) and an access structure $((\boldsymbol{A}^*, \boldsymbol{\rho}^*), \mathcal{T}_0^*)$, $((\boldsymbol{A}^*, \boldsymbol{\rho}^*), \mathcal{T}_1^*)$ in Phase 1 with the restriction that none of them can fulfill any of the queried attributes set S_i. In response, challenger $\mathcal{B}$ selects a bit $b \leftarrow \{0,1\}$, choose Q_0, $Q_x \in \mathbb{G}p4$ at random. The challenge ciphertext CT^* of the message M_b^* was then computed under the access structure $((\boldsymbol{A}^*, \boldsymbol{\rho}^*), \mathcal{T}^*)$, and the challenge ciphertext CT^* was sent to Adversary $\mathcal{A}$.

Phase 2: Repeat Phase 1. The adversary $\mathcal{A}$ request a private key, however none of the attribute S_{i} met the access structure.

Guess: If $b = b\prime$, the adversary $\mathcal{A}$ produces a guess bit $b' \in \{0,1\}$ and wins the game.

In this game, the adversary $\mathcal{A}$'s advantage is defined as $\left|\Pr[b = b'] - \frac{1}{2}\right|$, where the probability is divided by the number of random bits used by the adversary $\mathcal{A}$ and the challenger $\mathcal{B}$.

Definition 2. *A PHLC scheme with hidden access policy is fully secure if all polynomial-time adversaries have at most a negligible advantage in the security game.*

4. Implementation of Policy Hiding in CP-ABE Using Logical Connective

Note that the attribute values of the access policy contain sensitive users' data. For example, the medical and healthcare Cloud system's attribute value could include information on patients' ailments and family history of hereditary diseases. Hence, such information needs to be concealed to protect the users' privacy. Applying the attribute hiding in access policy preserves the attribute values for gaining Cloud data privacy.

In our encryption solution, the access policy is constructed in CP-ABE using the policy hiding logical connective (PHLC) strategy. Specifically, as mentioned earlier, we integrate policy hiding schemes into CP-ABE components, that is, Setup, Key Generation, Encryption, and Decryption. We improved the Data Owner (DO) roles in the CP-ABE by appointing the encryption process with a hiding policy. Figure 3 describes in detail our enforcement of hiding by enhancing CP-ABE.

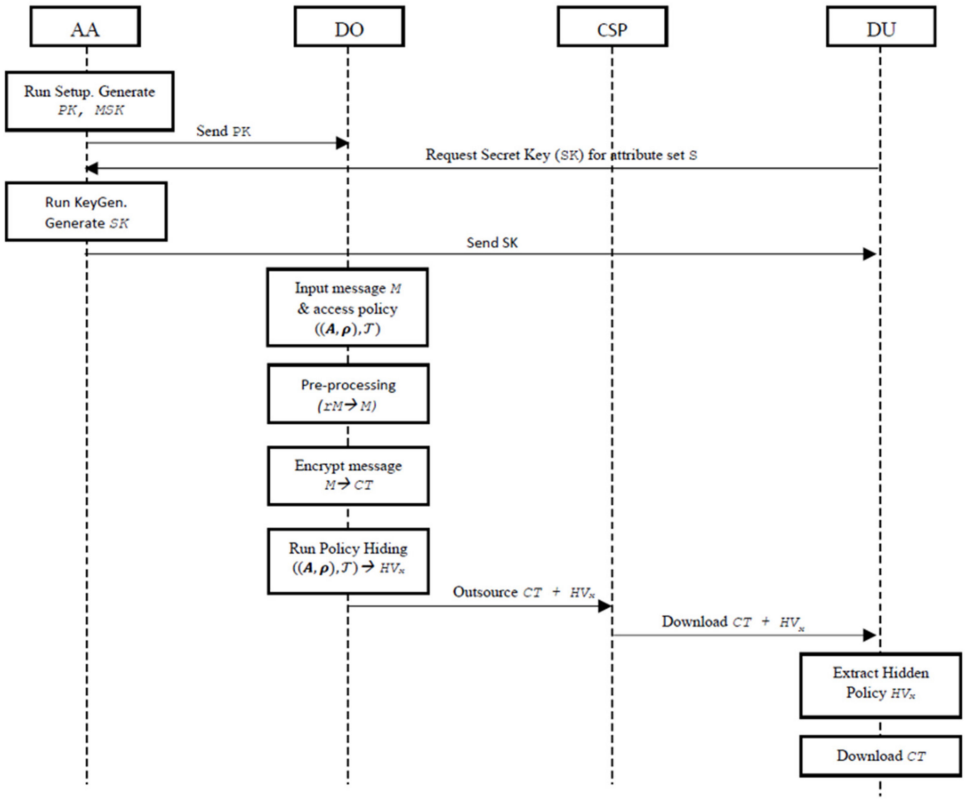

Figure 3. Process sequence chart of the proposed CP-ABE scheme.

4.1. SETUP $(1^{\lambda}) \rightarrow PK, MSK$

The Attribute Authority (AA) ran the setup algorithm by taking 1^{λ} as a security parameter in the setup module. The setup algorithm produced a tuple N = ($p1p2p3p4$; $\mathbb{G}$; $\mathbb{G}_T$; e) which contained four prime numbers, $p1, p2, p3, p4$. Meanwhile, four distinct ordered subgroups given as $\mathbb{G}p1$, $\mathbb{G}p2$, $\mathbb{G}p3$, $\mathbb{G}p4$ are structured based on the prime numbers. Let $\mathbb{G}$ and $\mathbb{G}_T$; be a cyclic group with order N. Therefore, the attribute authority uniformly chooses a, $\alpha, \alpha 1, \beta \in_R \mathbb{Z}_N$ and $g, g1 \in \mathbb{G}p$. H, H_1 were set as public hash functions, with H mapped the attribute value, AV_x to an element in $\mathbb{Z}_N$, and H_1 was a pseudo-random function that mapped elements in $\mathbb{G}$ and $\mathcal{M}$ to elements in $\mathcal{M}$. e is a bilinear map, and it will be computed value Y, Public Parameters PK, and Master Key MSK as:

$$Y = e(g,\ g_1)^{\alpha\alpha 1},$$

$$PK = \left\{N,\ g,\ g^{a}, g^{\alpha 1}, g^{\beta}, Y\right\},$$

$$MSK = \{a, \alpha, \alpha_1, \beta, g_1\}$$

4.2. KEYGEN (PK, MSK, S) → SK

Attribute Authority (AA) checked the users to ensure that only legitimate users could access the file system. The AA aborted any illegal access towards the file system by generating the secret key. Specifically, it takes input public parameters PK, master key MSK, users' attributes $S = (B_s,\ J_s)$ to generate the secret key, SK. Given B_s represents the attribute name index set and J_s is the attribute value set for the user. The AA ran the KeyGen module as follows:

AA chose $t \in R\mathbb{Z}_N$ and $R,\ R_1,\ R_i \in R\ \mathbb{G}p3$ for $i \in B_s$. It computed $K_1,\ K_2, K_i$ as

$$K_1 = g_1^{\alpha} g_1{}^{dt}.\ R,$$

$$K_2 = g_1{}^{t\alpha 1}.R_1,$$

$$K_i = \left(g_1{}^{H(L_i)} g_1{}^{\beta}\right)^t.R_i.$$

Then, the secret keys associated with attribute set $S = (B_s, J_s)$ were calculated as

$$SK = (K_1, K_2, \{K_i\}_{i \in B_s}.$$

4.3. PRE-PROCESSES (rM) → (M)

This new module is constructed to eliminate redundant data in a raw file before it has been encrypted in the encryption module. Each data in the input message is assigned an index number. For each re-appearance word, the index number is combined with the existing word, and the new message without redundant data is saved as message M. Algorithm 1 represents the pre-processing process named FWA pre-processing.

Algorithm 1 FWA pre-processing

Input: raw message ***rM***
Output: message M
Begin
$i = 0$, $M\varnothing$
Foreach $msg\ [i]$
If $msg\ [i] \neq EOF$
Search $msg\ [i]$ in M
If $msg\ [i]$ exist,
update M
Else
Add $msg\ [i]$ in M
$i++$
End the process

4.4. ENCRYPT $(\boldsymbol{PK}, \boldsymbol{M}, \mathbb{A}) \rightarrow \boldsymbol{CT}$

In this module, we input public parameter PK, Message M and access structure $\mathbb{A} = ((\boldsymbol{A}, \boldsymbol{\rho})\mathcal{T}))$ and produced the ciphertext, CT. In access structure $\mathbb{A}$, $\boldsymbol{A}$ is an access matrix $l \times n$ and $\boldsymbol{\rho}$ mapped each row $\boldsymbol{A}_x$ to an index of the attribute name. Meanwhile $\mathcal{T} = \left(t_{\rho(1)},\ t_{\rho(2)}, t_{\rho(3)}, \ldots, t_{\rho(l)}\right) \in\ \mathbb{Z}_N^l\left(t_{\rho(x)} \in AV_{\rho(x)}\right)$ is a set of attribute-value related to the access policy $(\boldsymbol{A},\ \boldsymbol{\rho})$. The encryption algorithm selected a random vector $V = (s, y_2, y_3, \ldots,\ y_n$ where $s,\ y_2,\ y_3,\ \ldots,\ y_n$ was randomly selected from $\mathbb{Z}_N$ and s is a shared value. For $x = 1$ to l, it computed $\lambda_x = \boldsymbol{A}_x \times V$, where $\boldsymbol{A}_x$ corresponded to the x^{th} row of $\boldsymbol{A}$ and calculated $X = \mathcal{E}_{Enc}(k,\ M')$, $\mathcal{F} = H_1(k \parallel M')$. Additionally, it also randomly took $Q_0, \{Q_x\}_{1 \leq x \leq 1 \in R} \mathbb{G}_{p4}$. Finally, it calculated the entire ciphertext components $C_0, C_1\{C_x\}_{1 \leq x \leq 1}$ as follows:

$$C_0 = ke(g, g_1)^{\alpha\alpha 1s},$$
$$C_1 = g^{s\alpha 1}.Q_0,$$
$$C_x = g^{a\lambda x}\left(g^{H(t\rho(x))}g^{\beta}\right)s.Q_x.$$

Previously, the access policy $((\boldsymbol{A}, \boldsymbol{\rho}), \mathcal{T})$ is appended to the ciphertext CT then outsourced to the cloud storage. However, this access policy is in a readable format, possibly exposing several sensitive information about the users. The researchers in [25] had emphasized that the attribute mapping function ρ will be caused attribute leakage. Hence, we improved the scheme to prevent the user's privacy leakage by eliminating the attribute mapping function. In this scheme, we replaced attribute value in access structure $\mathbb{A} = ((\boldsymbol{A}, \boldsymbol{\rho})\mathcal{T}))$ with attribute location in the form of (x, y). Nonetheless, this strategy is insufficient to preserve Cloud data privacy. Therefore, XOR-based logical connective is used in policy hiding strategy to enhance the privacy of access policy.

Our policy hiding logical connective (PHLC) strategy is used to convert the location attribute value $(\mathcal{T})$ in the access policy to ciphertext. We specifically extracted the exact location of attribute value from the access policy $((\boldsymbol{A}, \boldsymbol{\rho}), \mathcal{T})$. Once the location is obtained, it has then been converted into a ciphertext based on the XOR operation. Figure 4 portrays the example of the access matrix $Att_{(x,y)}$ where the $-x$ and $-y$ values are represented as the location of the attribute value, which comprises of attribute name, attribute value, and the act of mapping ρ to the index of the attribute name. We removed the mapping ρ and used the attribute location $(Att_{x,y})$.

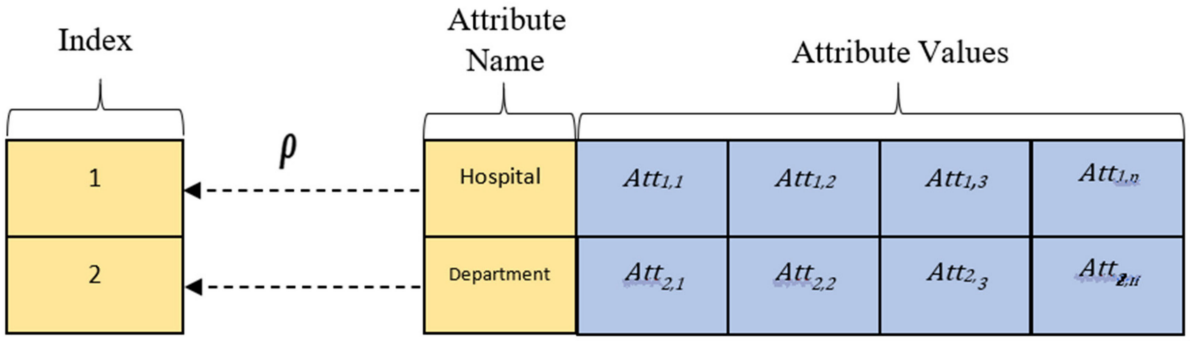

Figure 4. Example of the access. matrix $Att_{(x,y)}$.

The post-encryption module is then executed where it is where our main contribution of CP-ABE extension takes place. It is deliberated as below:

4.5. Policy Hiding $((A, \rho), \mathcal{T}) \rightarrow HV$

In this process, we derived the hidden value HV, which is the location of the attribute value that has been encrypted. The policy hiding algorithm adopts an access policy $((\boldsymbol{A}, \boldsymbol{\rho}), \mathcal{T})$ as an input. It first extracted the attribute values' set associated with access policy $(\mathcal{T})$ and then got the exact location (x, y) of each attribute value. Each location is converted to ciphertext via operation $\oplus$ and $\odot$. Finally, our CP-ABE solution produced the output of ciphertext and hidden value in (CT, HV) and outsourced it to Cloud servers. Algorithm 2 depicts the entire Policy Hiding Algorithm.

Algorithm 2 Policy Hiding Algorithm

Input: Access policy $((\boldsymbol{A}, \boldsymbol{\rho}), \mathcal{T})$
Output: Hidden Access Policy (HV)
Begin
foreach AV_x *of* $\mathcal{T}$, $\forall_x$ = index of *AV.*
Extract location (X_x, Y_x)
Convert location (X_x, Y_x) String to Binary
Compute $\alpha_x = X_x \oplus Y_x$
Compute $\beta_x = \alpha_x \odot Y_x$
Convert Binary to Hexadecimal for each β_x, store as = HV_x
End the process

4.6. Decryption

Data users (recipients) accessed the encrypted data from the Cloud and downloaded it according to their preferences. Nonetheless, the decryption process is secured via access controls in which only legitimate data users are allowed to decrypt the ciphertext. Hence, when the user attribute $S = (B_s, J_s)$ satisfied with the access policy; the users are eligible to perform the decryption process. In prior, the data user (DU) needs to extract the hidden policy after receiving $CT = (C_0, C_1\{C_x\}_{1 \le x \le 1}, HV)$ from the Cloud storage based on the following decryption algorithm.

4.7. Ext Hidden Policy $(\boldsymbol{CT}, \boldsymbol{HV}) \rightarrow (\mathcal{T})$

In the decryption process, the hidden value, HV is used as an input that is retrieved from the cloud storage. Based on HV, it converted the value of HV into a binary set, then formed the operation of $\oplus$ and $\odot$ obtained the attribute's original location. It describes further in Algorithm 3.

Algorithm 3 Extracting Hidden Access Policy

Input: Hidden Access Policy (HV)
Output: Access policy $((\boldsymbol{A}, \boldsymbol{\rho}), \mathcal{T})$
Begin
foreach HV_x of access policy $\forall_x$ = index of HV.
Convert hexadecimal HV_x to binary; store as (X_x, Y_x)
Compute $\delta_x = X_x \oplus Y_x$
Compute $\boldsymbol{\Omega}_x = \delta_x \odot Y_x$
Convert $\boldsymbol{\Omega}_x$ into Unicode Text
End the process

Upon successful concealment of the hidden access policy, the data users run the following decryption algorithm below:

4.8. Decrypt $(\boldsymbol{PK}, \boldsymbol{SK}, \boldsymbol{CT}, \boldsymbol{S}) \rightarrow M$

Similar to [23], after accepting CT, the decrypt algorithm checks whether the hash value of $H(J_s) = H\left(t_{\rho(x)}\right)$. If the value is equal, then the system authorized the DU to decrypt the CT as below:

$$E = e(g, g_1)^{\alpha\alpha_1 s}$$

$$k = C_o / E$$

Given that if the key k of the symmetric encryption scheme had been successfully computed, then the decryption algorithm determined the values $F' = H_1(k \parallel M')$. Only if the equation $F = F'$ holds, the message M will be produced. Otherwise, the process will be terminated. The plaintext is recovered via the calculation $M = \mathcal{E}_{Dec}(k, x)$. Specifically, the reinforcement hiding in access policy is employed to maintain the Cloud data privacy by concealing the values of attributes. The policy hiding (Algorithm 2) is concealed in the

encryption module, and it runs after the encryption algorithm. Algorithm 3 then extracted the policy hiding algorithm before the decryption algorithm.

5. Security Analysis

In this section, we have extended our investigation to security analysis. We focused on policy privacy preservation and designed the security analysis in the following discussion.

5.1. Security Proof

This section discusses the security proof of the proposed scheme. Our proposed scheme is constructed by re-simulating the CP-ABE scheme published in [23], which has been proved to be secure by attaining full security in the standard model utilizing the dual system encryption approach under static assumptions. Therefore, we provide a security analysis of the enhanced scheme on hidden access policy presented in Theorem 1. While in theorem 2 we discussed our scheme against traffic analysis.

Theorem 1. *PHLC preserves the privacy of access policy against the polynomial-time adversary in the security parameter* λ.

Proof. In the PHLC scheme, the attribute's location in the access policy is converted into a hidden value using X-OR operation. So, this hidden access policy stored in Cloud together with $CT = (C_0, C_1\{C_x\}_{1 \leq x \leq 1}, HV))$. DU with attributes set s that satisfied access structure $\mathbb{A} = ((\boldsymbol{A}, \boldsymbol{\rho})\mathcal{T}))$ can decrypt CT. The adversary $\mathcal{A}$ who has no knowledge about the scheme used to convert the hidden access policy could not launch the brute force attack to guess the attribute string within polynomial time. Furthermore, they are unable to sniff any sensitive data from the modified access policy established as HVx. DU is only permitted to validate their attributes in the hidden access policy, and it is forbidden to inspect any attributes in the attribute universe unless they collude with others. □

Theorem 2. *PHLC secures against Traffic Analysis.*

Proof. In our scheme, the ***CT*** is stored in Cloud storage together with HV. We assume that an adversary $\mathcal{A}$ successfully analyses the packet during the transmission to the Cloud and gain access to ciphertext and hidden value ($CT = (C_0, C_1\{C_x\}_{1 \leq x \leq 1}, HV)$), thus the adversary $\mathcal{A}$ could not read the message because it is in the unreadable format (ciphertext). In this case, we assume that the symmetric encryption key based on AES is secured. In addition, the adversary $\mathcal{A}$ also could not gather any information from the access policy because the access policy is in the ciphertext. Hence it is intractable to compute the access policy. □

5.2. Case Scenario Security Analysis Simulation

To prove the compelling of our CP-ABE enhancement solution against traffic analysis, we conducted a case study, as shown in Figure 5. In this experiment, we conducted a passive attack that attempted to learn the system's information without affecting the system's resources. We employed FileZilla Transfer Protocol (FTP) Client and Server Tools for transferring data between the owner and user, as shown in Figure 6. As an adversary in this experiment, the attacker had been designed to perform an unauthorized sniff and read the information through the access policy.

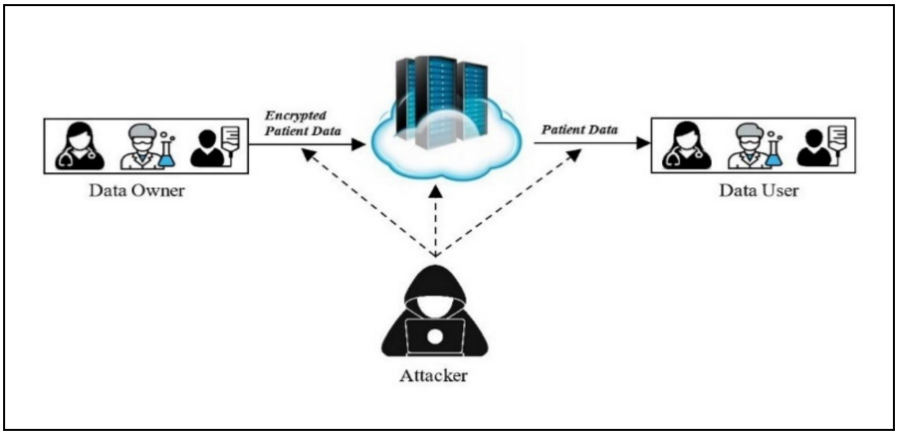

Figure 5. Passive Attack Architecture.

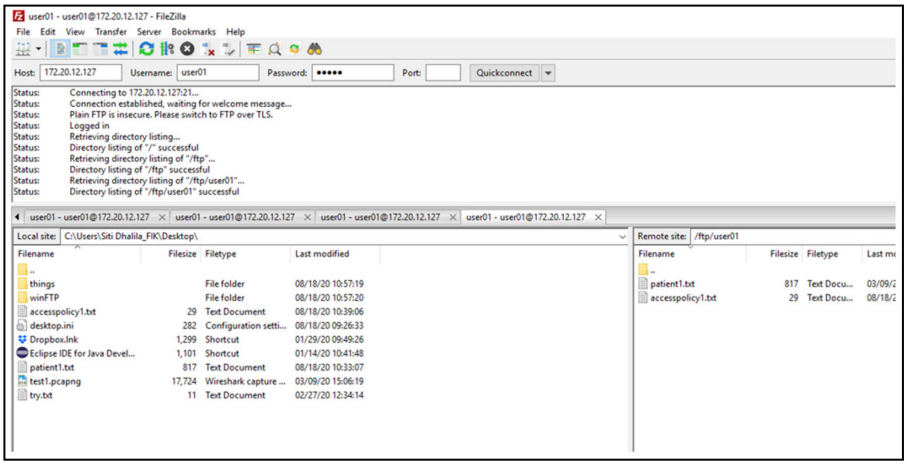

Figure 6. Filezilla Tool.

In this experiment, the FileZilla Client had been installed on the DO computer. The file is transferred via FTP upon a successful connection between the DO s computer and the DU. The adversary $\mathcal{A}$ exploited the communication between the DO and DU via the Wireshark packet analysis tool by sniffing the inter-communication and capture the data (in packet form).

We experimented with two scenarios. The first scenario involved the DO sending the ciphertext $\boldsymbol{CT}$ with access policy $((\boldsymbol{A}, \boldsymbol{\rho}), \mathcal{T})$ in a readable format to the DU, as shown in Figure 7. While Figure 8 discovers the second scenario where the packet of the ciphertext had been applied with the Policy Hiding Algorithm and sent to the DU. The packet shown in Figure 8 are in unreadable format (HV).

Based on these results, the study proved that the adversary $\mathcal{A}$ unable to retrieve any information regarding the data because it is in unreadable (ciphertext) format. Additionally, the adversary $\mathcal{A}$ could not learn anything because the policy is also generated in scrambled form. This experiment shows that our proposed scheme resists any malicious act to sniff the information during the data transmission.

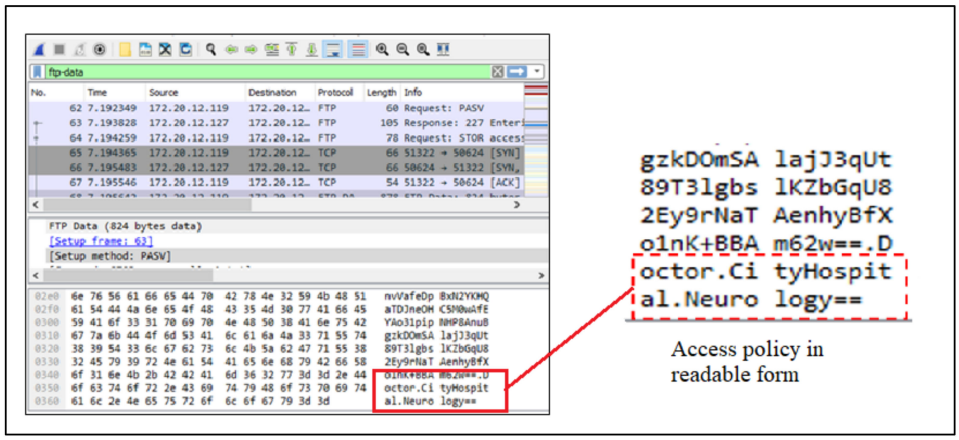

Figure 7. Packet capturing using Wireshark.

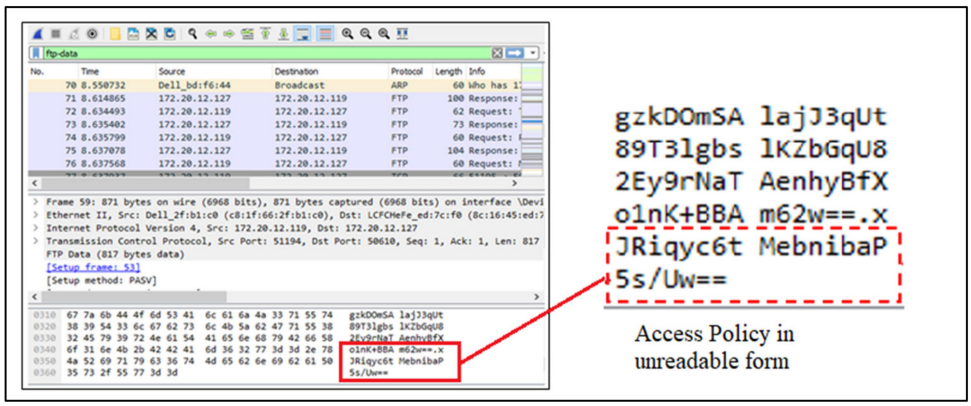

Figure 8. Packet capture using Wireshark after access policy hiding.

6. Result and Discussion

In this section, we addressed the experimental setting, performance evaluation and discussed the result.

6.1. Experiment Configuration

We developed our proposed scheme based on the Java Pairing-based Cryptography library (JPBC) in the Eclipse IDE for Java developers in 2019-03. In this experiment, the number of attributes used varies from 2 to 14, based on the author's experimental settings [23]. Our work placed Zhang's [23] research work as the benchmark because Zhang's work is closely related to our policy hiding approach, which focused on preserving Cloud data privacy. In addition, we also compared our scheme with the expressive CP-ABE scheme proposed by [33]. Similar to our scheme, both CP-ABE in [23,33] also employed LSSS to express the access structure and exhibited the same Cloud storage system architecture.

6.2. Result

In this section, we compared our scheme's performance regarding the storage cost of ciphertext and encryption time with previous works [23,33]. Figure 9 shows the ciphertext size for our PHLC-based CP-ABE scheme, Zhang's work [23,33]. As illustrated in this figure, our proposed scheme is comparable with the with Zhang's work and [33]. This figure indicates that our data privacy strategy realizes effective ciphertext size even though the PHLC scheme hide both attribute names and attribute values and attached it to the ciphertext. While in Figure 10 illustrates the time taken for the Data Owner (DO) to encrypt the file before outsourcing it to the Cloud storage. Based on the figure, it shows that

the encryption time exhibits a linear increase with the number of attributes. Further, the PHLC encryption time is less than in Zhang's work [23], on average 20%. Even though the difference is slight, it greatly impacts the processing overhead and complexity of the encryption process. The data privacy approach by [33] showed the maximum encryption time, which can be improved to gain effective encryption time.

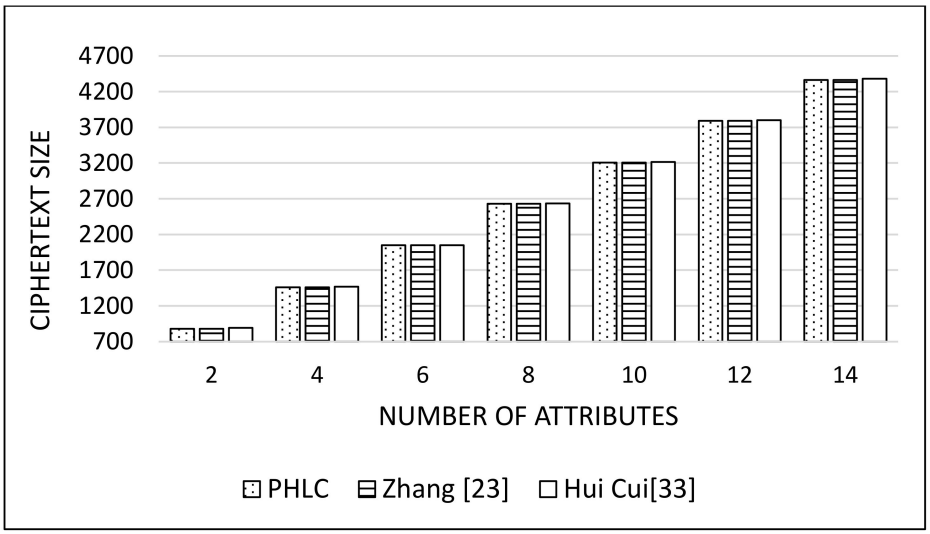

Figure 9. Storage cost of ciphertext.

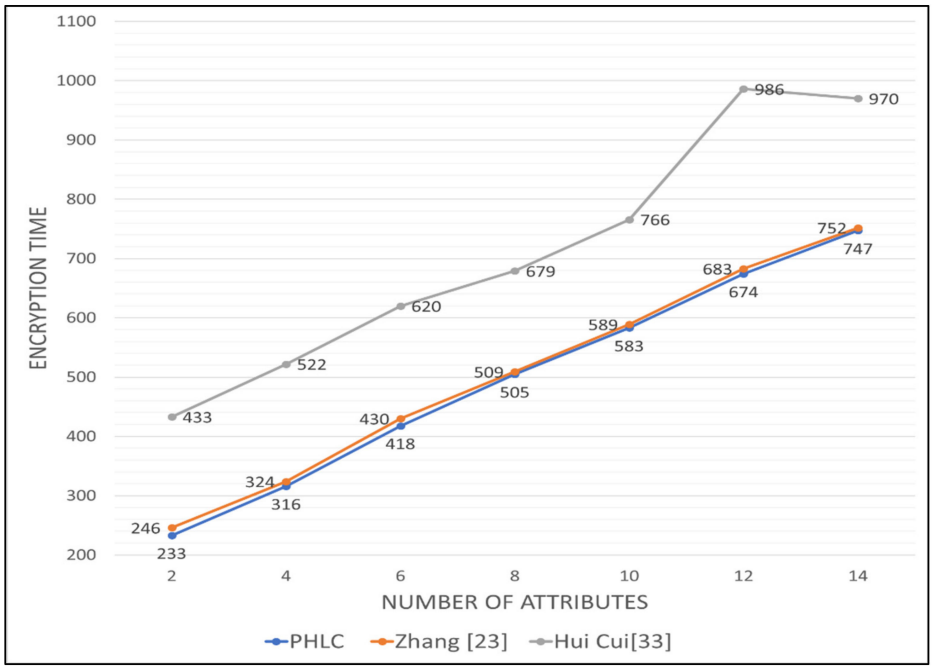

Figure 10. Encryption Time.

To reveal the effectiveness of our PHLC approach in terms of ciphertext size, we enhanced the experiment by adding a pre-processing module. This module eliminates redundant data in a raw shared file using a new technique called frequent wording appearance pre-processing. Based on Figure 11, the ciphertext size in PHLC after applying the pre-processing module have decreased by approximately 17% on average. Reducing the size of the ciphertext helps reduce the storage space and leads to efficient decryption process performance.

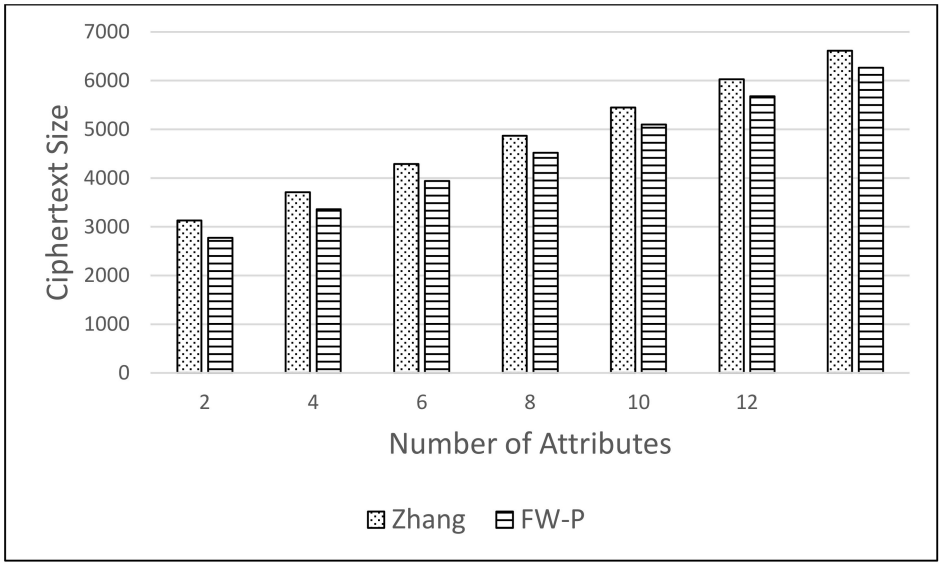

Figure 11. Storage cost of ciphertext.

7. Conclusions

In the ever-increasing era of data breaches, providing the integrity and privacy of Healthcare Cloud storage is a challenging issue in the Cloud environment, especially when the demands and requests for Cloud services are increasing. Cloud computing must deliver a security solution that protects sensitive information and data sharing processes. This solution can prevent the third party from eavesdropping or tampering with the data while it is being transmitted. Therefore, this work enhances the CP-ABE approach by proposing two new modules: the pre-processing and the fully hidden access policy modules. In addition to performing the encryption algorithm, the data attributes in the access policy are further concealed. It makes the unauthorized party unable to learn any details about the access policy and encrypted data. The simulation results demonstrated that our CP-ABE preserved the Cloud data without incurring large sizes of ciphertext, compelling encryption time. As in the case of the passive attack, the security analysis also revealed that the enhanced CP-ABE is secure enough to be executed in the real environment. Therefore, Cloud data privacy has become an essential feature in multi-tenant computing environments where any form of data disruption is unacceptable.

Author Contributions: Conceptualization, S.D.M.S., M.H., Z.M.H. and M.A.M.; Methodology, S.D.M.S., M.H., Z.M.H. and M.A.M.; Software, S.D.M.S.; Validation, S.D.M.S., M.H. and M.A.M.; Writing—original draft, S.D.M.S.; Writing—review & editing, M.H., Z.M.H. and M.A.M. All authors have read and agreed to the published version of the manuscript.

Funding: This work was support by UPM Journal Publication Fund, Universiti Putra Malaysia and CREIM, Universiti Sultan Zainal Abidin.

Data Availability Statement: Not Applicable, the study does not report any data.

Acknowledgments: The authors would like to thank Universiti Sultan Zainal Abidin and Universiti Putra Malaysia.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Al Hamid, H.A.; Rahman, S.M.M.; Hossain, M.S.; Almogren, A.; Alamri, A. A Security Model for Preserving the Privacy of Medical Big Data in a Healthcare Cloud Using a Fog Computing Facility with Pairing-Based Cryptography. *IEEE Access* **2017**, *5*, 22313–22328. [CrossRef]

2. Kumar, P.; Alphonse, P.J.A. Attribute based encryption in cloud computing: A survey, gap analysis, and future directions. *J. Netw. Comput. Appl.* **2018**, *108*, 37–52. [CrossRef]
3. Kajiyama, T.; Jennex, M.; Addo, T. To cloud or not to cloud: How risks and threats are affecting cloud adoption decisions. *Inf. Comput. Secur.* **2017**, *25*, 634–659. [CrossRef]
4. Jamal, F.; Abdullah, M.T.; Hanapi, Z.M.; Abdullah, A. Reliable Access Control for Mobile Cloud Computing (MCC) With Cache-Aware Scheduling. *IEEE Access* **2019**, *7*, 165155–165165. [CrossRef]
5. Ramachandra, G.; Iftikhar, M.; Khan, F.A. A Comprehensive Survey on Security in Cloud Computing. *Procedia Comput. Sci.* **2017**, *110*, 465–472. [CrossRef]
6. Zhang, L.; Cui, Y.; Mu, Y. Improving Security and Privacy Attribute Based Data Sharing in Cloud Computing. *IEEE Syst. J.* **2019**, *14*, 387–397. [CrossRef]
7. Yang, J.-J.; Li, J.-Q.; Niu, Y. A hybrid solution for privacy preserving medical data sharing in the cloud environment. *Future Gener. Comput. Syst.* **2015**, *43–44*, 74–86. [CrossRef]
8. Satar, S.D.M.; Hussin, M.; Hanapi, Z.M.; Mohamed, M.A. Data Privacy and Integrity Issues Scheme in Cloud Computing: A Survey. *Int. J. Eng. Technol.* **2018**, *7*, 102–105. [CrossRef]
9. Yan, Z.; Deng, R.H.; Varadharajan, V. Cryptography and Data Security in Cloud Computing. *Inf. Sci.* **2017**, *387*, 53–55. [CrossRef]
10. Somani, U.; Lakhani, K.; Mundra, M. Implementing digital signature with RSA encryption algorithm to enhance the Data Security of cloud in Cloud Computing. In Proceedings of the 2010 First International Conference On Parallel, Distributed and Grid Computing (PDGC 2010), Solan, India, 28–30 October 2010; pp. 211–216. [CrossRef]
11. Lai, J.; Deng, R.H.; Li, Y. Expressive CP-ABE with partially hidden access structures. In Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, Seoul, Korea, 2–4 May 2012; pp. 18–19. [CrossRef]
12. Rizk, R.; Alkady, Y. Two-phase hybrid cryptography algorithm for wireless sensor networks. *J. Electr. Syst. Inf. Technol.* **2015**, *2*, 296–313. [CrossRef]
13. Singh, A.P.; Pasupuleti, S.K. Optimized Public Auditing and Data Dynamics for Data Storage Security in Cloud Computing. *Procedia Comput. Sci.* **2016**, *93*, 751–759. [CrossRef]
14. Potey, M.M.; Dhote, C.; Sharma, D.H. Homomorphic Encryption for Security of Cloud Data. *Procedia Comput. Sci.* **2016**, *79*, 175–181. [CrossRef]
15. Satar, S.D.M.; Hussin, M.; Hanapi, Z.M.; Mohamed, M.A. Cloud-Based Secure Healthcare Framework by using Enhanced Ciphertext Policy Attribute-Based Encryption Scheme. *Int. J. Adv. Comput. Sci. Appl.* **2021**, *12*, 393–399.
16. Singh, N.; Singh, A.K. Data Privacy Protection Mechanisms in Cloud. *Data Sci. Eng.* **2017**, *3*, 24–39. [CrossRef]
17. Sabitha, S.; Rajasree, M. Access control based privacy preserving secure data sharing with hidden access policies in cloud. *J. Syst. Arch.* **2017**, *75*, 50–58. [CrossRef]
18. Sahai, A.; Waters, B. Fuzzy Identity-Based Encryption. *Lecture Notes in Computer Science*. 2005. Available online: https://eprint.iacr.org/2004/086.pdf (accessed on 15 October 2021).
19. Bethencourt, J.; Sahai, A.; Waters, B. Ciphertext-policy attribute-based encryption. In Proceedings of the 2007 IEEE Symposium on Security and Privacy (SP'07), Berkeley, CA, USA, 20–23 May 2007; pp. 321–334.
20. Muhammad, N.; Zain, J.M.; Mohamad, M. Current Issues in Ciphertext Policy-Attribute Based Scheme for Cloud Computing: A Survey. *Int. J. Eng. Technol.* **2018**, *7*, 64. [CrossRef]
21. Liu, X.; Xia, Y.; Yang, W.; Yang, F. Secure and efficient querying over personal health records in cloud computing. *Neurocomputing* **2018**, *274*, 99–105. [CrossRef]
22. Khuntia, S.; Kumar, P.S. New Hidden Policy CP-ABE for Big Data Access Control with Privacy-preserving Policy in Cloud Computing. In Proceedings of the 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Bengaluru, India, 10–12 July 2018; pp. 1–7. [CrossRef]
23. Zhang, L.; Hu, G.; Mu, Y.; Rezaeibagha, F. Hidden Ciphertext Policy Attribute-Based Encryption with Fast Decryption for Personal Health Record System. *IEEE Access* **2019**, *7*, 33202–33213. [CrossRef]
24. Li, C.; Zhang, Y.; Xie, E.Y. When an attacker meets a cipher-image in 2018: A year in review. *J. Inf. Secur. Appl.* **2019**, *48*. [CrossRef]
25. Xu, S.; Yang, G.; Mu, Y. Revocable attribute-based encryption with decryption key exposure resistance and ciphertext delegation. *Inf. Sci.* **2018**, *479*, 116–134. [CrossRef]
26. Ying, Z.; Wei, L.; Li, Q.; Liu, X.; Cui, J. A Lightweight Policy Preserving EHR Sharing Scheme in the Cloud. *IEEE Access* **2018**, *6*, 53698–53708. [CrossRef]
27. Zhang, Z.; Li, C.; Gupta, B.B.; Niu, D. Efficient Compressed Ciphertext Length Scheme Using Multi-Authority CP-ABE for Hierarchical Attributes. *IEEE Access* **2018**, *6*, 38273–38284. [CrossRef]
28. Fu, X.; Nie, X.; Wu, T.; Li, F. Large universe attribute based access control with efficient decryption in cloud storage system. *J. Syst. Softw.* **2018**, *135*, 157–164. [CrossRef]
29. Nishide, T.; Yoneyama, K.; Ohta, K. Attribute-Based encryption with partially hidden encryptor-specified access structures. In Proceedings of the International Conference on Applied Cryptography and Network Security, New York, NY, USA, 3–6 June 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 111–129.
30. Zhou, Z.; Huang, D.; Member, S.; Wang, Z. Efficient Privacy-Preserving Ciphertext-Policy Attribute Based-Encryption and Broadcast Encryption. *IEEE Trans. Comput.* **2015**, *64*, 126–138. [CrossRef]

31. Tamizharasi, G.S.; Balamurugan, B.; Gaffar, H.A. Privacy preserving ciphertext policy attribute based encryption scheme with efficient and constant ciphertextsize. In Proceedings of the 2016 International Conference on Inventive Computation Technologies, Coimbatore, India, 26–27 August 2016; Volume 3, pp. 1–5. [CrossRef]
32. Hu, G.; Zhang, L.; Mu, Y.; Gao, X. An Expressive "Test-Decrypt-Verify" Attribute-Based Encryption Scheme with Hidden Policy for Smart Medical Cloud. *IEEE Syst. J.* **2020**, *15*, 365–376. [CrossRef]
33. Cui, H.; Deng, R.H.; Lai, J.; Yi, X.; Nepal, S. An efficient and expressive ciphertext-policy attribute-based encryption scheme with partially hidden access structures, revisited. *Comput. Netw.* **2018**, *133*, 157–165. [CrossRef]
34. Xiong, H.; Zhao, Y.; Peng, L.; Zhang, H.; Yeh, K.-H. Partially policy-hidden attribute-based broadcast encryption with secure delegation in edge computing. *Future Gener. Comput. Syst.* **2019**, *97*, 453–461. [CrossRef]
35. Zhang, Y.; Zheng, D.; Deng, R.H. Security and Privacy in Smart Health: Efficient Access Control. *IEEE Internet Things J.* **2018**, *5*, 2130–2145. [CrossRef]
36. Zhang, Y.; Chen, X.; Li, J.; Wong, D.S.; Li, H.; You, I. Ensuring attribute privacy protection and fast decryption for outsourced data security in mobile cloud computing. *Inf. Sci.* **2017**, *379*, 42–61. [CrossRef]
37. Waters, B. Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In *International Workshop on Public Key Cryptography*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6571, pp. 53–70.
38. Rosen, K.H. Discrete Mathematics and Its Applications. Available online: http://site.iugaza.edu.ps/lalsaedi/files/2012/01/Kenneth_H._Rosen_-_Discrete_Mathematics_and_Its_Applications.pdf (accessed on 15 October 2021).
39. Shyu, S.J. XOR-Based Visual Cryptographic Schemes with Monotonously Increasing and Flawless Reconstruction Properties. *IEEE Trans. Circuits Syst. Video Technol.* **2017**, *28*, 2397–2401. [CrossRef]