mathematics

Article

Ciphertext-Policy Attribute-Based Encryption for Cloud Storage: Toward Data Privacy and Authentication in AI-Enabled IoT System

P. Chinnasamy [1], P. Deepalakshmi [2], Ashit Kumar Dutta [3], Jinsang You [4,*] and Gyanendra Prasad Joshi [5,*]

1 Department of Computer Science and Engineering, MLR Institute of Technology, Hyderabad 500043, India; chinnasamyponnusamy@gmail.com
2 Department of Computer Science and Engineering, Kalasalingam Academy of Research and Education, Srivilliputtur 626128, India; deepa.kumar@klu.ac.in
3 Department of Computer Science and Information Systems, College of Applied Sciences, AlMaarefa University, Ad Diriyah, Riyadh 13713, Saudi Arabia; adotta@mcst.edu.sa
4 Seculayer Company, Ltd., Seoul 04784, Korea
5 Department of Computer Science and Engineering, Sejong University, Seoul 05006, Korea
* Correspondence: js.yu@seculayer.com (J.Y.); joshi@sejong.ac.kr (G.P.J.); Tel.: +82-2-6935-2481 (G.P.J.)

Abstract: People can store their data on servers in cloud computing and allow public users to access data via data centers. One of the most difficult tasks is to provide security for the access policy of data, which is also needed to be stored at cloud servers. The access structure (policy) itself may reveal partial information about what the ciphertext contains. To provide security for the access policy of data, a number of encryption schemes are available. Among these, CP-ABE (Ciphertext-Policy Attribute-Based Encryption) scheme is very significant because it helps to protect, broadcast, and control the access of information. The access policy that is sent as plaintext in the existing CP-ABE scheme along with a ciphertext may leak user privacy and data privacy. To resolve this problem, we hereby introduce a new technique, which hides the access policy using a hashing algorithm and provides security against insider attack using a signature verification scheme. The proposed system is compared with existing CP-ABE schemes in terms of computation and expressive policies. In addition, we can test the functioning of any access control that could be implemented in the Internet of Things (IoT). Additionally, security against indistinguishable adaptive chosen ciphertext attacks is also analyzed for the proposed work.

Keywords: artificial intelligence; CP-ABE; chosen ciphertext attack; fine-grained access control; hiding access policy; SHA1

Citation: Chinnasamy, P.; Deepalakshmi, P.; Dutta, A.K.; You, J.; Joshi, G.P. Ciphertext-Policy Attribute-Based Encryption for Cloud Storage: Toward Data Privacy and Authentication in AI-Enabled IoT System. *Mathematics* **2022**, *10*, 68. https://doi.org/10.3390/math10010068

Academic Editor: Todor Tagarev

Received: 22 November 2021
Accepted: 21 December 2021
Published: 26 December 2021

1. Introduction

The Internet of Things (IoT) offers a new framework for the creation of heterogeneous and distributed networks and has become an increasingly ubiquitous forum for computing services. Over the last decade, several IoT applications have been introduced over various areas, including building and home automation, infrastructure and health care, transportation, manufacturing, and environmental monitoring [1–3]. Nonetheless, due to the lack of adequate computing and storage resources for processing and storing large volumes of IoT data, it appears to follow a cloud-based architecture to address the security and resource sharing problems. Therefore, the entire implementation infrastructure must be secured from attacks that can obstruct IoT services and pose a threat to data privacy, credibility, or confidentiality.

The most widely used method for maintaining users' confidentiality in the cloud environment is encryption. However, security problems dominate IoT-based applications [3], prompting a major overhaul of established security choices leading many researchers to the evolution of modern techniques. Some of the leading security issues are user access control

and maintaining the protection of cloud data. Within the following articles, we highlight some of the latest IoT and cloud computing approaches about access control systems, data protection, and privacy.

Cloud computing is among the most prominent IT technologies that have gained greater attention from government and industry since 2007. The cloud is composed of Service-Oriented Architecture (SOA), virtualization, variety of services, and deployment architecture [4]. It offers services in a pay-as-you-use procedure. It provides many features like cost, scalability, on-demand access to use the resources effectively. Though the users know about cloud advantages, the challenging task is to provide strong security and storage mechanisms due to its internet-based storage and organization of data. Data to be stored in the cloud are in general sensitive and confidential, for example, Medical data and Military information [4]. This requires increased security mechanisms to improve data privacy as well as strong mechanisms for authentication and access control.

Before outsourcing to the cloud, user data should be encrypted to achieve enhanced data security [5]. To encrypt user data, many algorithms are proposed and implemented. The more preferable cryptographic algorithms are public key and secret key algorithms, which keep multiple copies of the same files whenever the data is shared among multiple users. The main issue here is when the key is obtained, all the protected files get leaked out. It is essential to produce the secret key for each user present in the data-sharing mechanism as a way to overcome this. The major concerns in access control-based data outsourcing are key management, and distribution [6]. There are several access control mechanisms developed since the 1960s [7]. Amongst these, Bell-la-palda [8] and Biba [8] are well-known access models. Access control models can be implemented in different ways, each with a different scope of control as well as a different set of operations and resources.

Attribute-based encryption (ABE) [9] provides desirable solutions to access control problems. The two important variants in the ABE scheme are KP-ABE (Key-Policy Attribute-Based Encryption) [10], whereby access policy is defined based on user's keys as well as attributes used to encrypt the data and CP-ABE [11,12], in which the attributes are related with user's secret key and access policies are related with the ciphertext. Compared to KP-ABE [10], and fuzzy identity-based encryption [13], CP-ABE [14] is more appropriate in offering an efficient fine-grained access control. The access policy is embedded within the ciphertext in CP-ABE. However, the CP-ABE ciphertext is considered to be decrypted by all users. Therefore, there may be a chance to carry out the malicious activity by the authorized user, leading to the existence of insider attack [11,15]. To avoid this type of attack, the only option is to verify the data owner's authentication. The most challenging task in the cloud would be detecting and preventing insider attacks because most of the records stored in the cloud are often sensitive. The Kandias et al. [15] stated that people who served inside their working organization carry out 85% fraud. Approximately the insider theft reduces 5% of the annual revenue of an organization, and almost 330 cases of insider theft are recognized in 2010. Protecting the privacy of access policy and also providing security against insider attacks is therefore necessary.

1.1. Motivation

With an example, Figures 1 and 2 thoroughly showed the notion of hidden access policy. In this case, the data owner should encrypt the data, which may then be accessed using the public access policy (see Figure 1), while the confidential data is protected using the anonymized access policy technique (i.e., Figure 2).

Traditional CP-ABE methods have several benefits, such as confidentiality, authentication, and access control, although certain issues with access policy privacy, information security, malicious insiders, storage complexity, and interoperability with IoT enabled infrastructures.

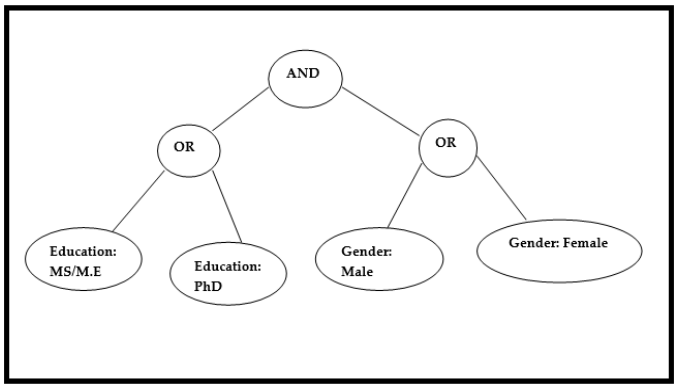

Figure 1. Public access policy.

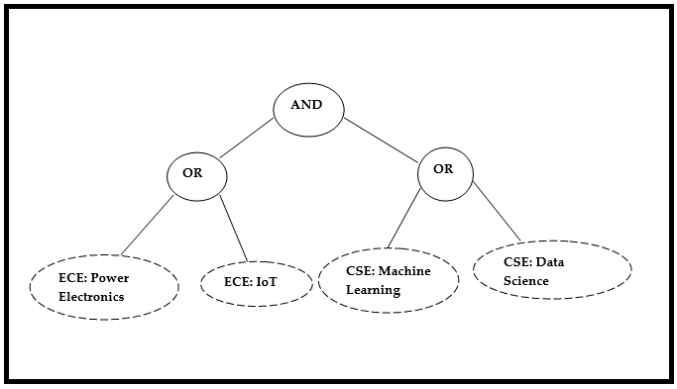

Figure 2. Fully hidden access policy: the information that is hidden is represented by the dotted node.

1.2. Contributions

Based on the foregoing observations, the following are the key contributions of this article:

(i) We propose a new strategy CP-ABE scheme to enhance the security of user data and privacy of the user by hiding the access policy.
(ii) The storage overhead of this scheme is reduced by generating constant size ciphertext.
(iii) The identification and prevention from insider attack is achieved by utilizing the short signature scheme.
(iv) An enhancement to the IoT architecture has been designed in addition to making it a secure fine-grained access control system that also prevents insider attacks.
(v) A comparison study was conducted to describe the key components of ABE schemes that hide access policies, along with computational overhead analysis, security, and operational attributes of various access control schemes.

1.3. Paper Organization

Rest of the paper is structured as follows. Section 2 describes the merits and demerits of the existing access control mechanism while Section 3 discusses the proposed method's mathematical background. Section 4 describes the proposed scheme's system model and Section 5 discusses the explanation of the proposed method. Section 6 deals with the implementation and performance assessment of the proposed method. Section 7 discusses the proposed method's security analysis and the conclusion is finally stated in Section 8.

2. Related Works

As related works, we hereby mention works on Attribute-Based Encryption(ABE), Predicate Encryption (PE), Ciphertext-Policy Attribute-Based Encryption (CP-ABE) and CP-ABE with hidden access structure.

As an option to attain privacy and fine-grained access control, Attribute-Based Encryption (ABE) [16] has been presented by Sahai and Waters [13] where the encryption is based on both access policy and private keys. They made ciphertext to be associated with set of attributes. In these methods, the ciphertext size, encryption and decryption times vary linearly with respect to the complexity of access formula.

The inner product Predicate Encryption (IPE) is the basis of CP-ABE since the security mechanisms correlate to predicates in PE and attributes is correlated for ciphertext and Katz et al. [17] implemented it. Pallavi [18] et al. presented a new CP-ABE method which supports hidden access policy. They used inner product encryption along with attribute hiding to provide unlinkability as well as to improve the patient data privacy. TVX Phuong [19] et al. presented a unique CP-ABE scheme based on two different processes. The first one is used to achieve only the constant ciphertext size. The second process is used to hide the access policy using Inner Product Encryption (IPE). However, the size of the ciphertext is dynamically changed. K. Frikken [20] et al. introduced new protocols to preserve sensitive credentials as well as sensitive policies. They used three different protocols to hide the credential information from intruder. X Yao [21] et al. introduced an anonymous based access control for the clouds. This method is not applicable for other applications because it is designed only for ciphertext. The cost of this scheme is high on user side.

Bethencourt [22] et al. initiated the CP-ABE scheme that provided security against collusion attacks. Subsequently, Doshi and Jinwala [23] et al. offered a novel CP-ABE scheme considered to be fully secure under attacks by Chosen Plaintext Attack (CPA) and Chosen Ciphertext Attack (CCA). The multi-authority ABE scheme has recently been studied in [24,25].

Helil [26] et al. presented an innovative CP-ABE scheme with hidden attributes. This is the novel scheme to define the sensitive data sets (SDS) constraint. Here, they used Chinese wall security policy to construct the SDS constraint. Compared to existing methods, this method takes extra communication cost due to SDS constraint. The access control policy and the restriction set are divided and partial information is fed to the proxy server and SDS monitor to prevent commercial errors. Sabitha [27] et al. introduced a new approach that preserves privacy through secure cloud hidden access policies for data sharing. Among these two [26,27] methods, first method is used to improve the privacy of user data and second one to prevent the insider attack. Compared to other attribute-based methods, this method has higher space complexity. Lixian Liu [28] et al. adopted partially hidden access structure in their proposal. This method was mainly used to improve the privacy of electronic medical record system. Here also, since the ciphertext size is linear, space complexity is high. Balu [29] et al. submitted a hidden access policy CP-ABE scheme by providing security using the assumption of Decisional Diffie-Hellman (DDH). Yadav [30] et al. introduced a novel scheme which hides the access policy in ABE. Hiding the access policy is a significant contribution of certain applications like Medical, Military communications etc. The partial information of ciphertext is mainly hidden from user in order to improve the privacy. Zhong [31] et al. introduced a decentralized multi-authority CP-ABE schemeagain using hidden access policy. The communication and computation cost of this scheme is low. Yang [32] et al. presented a mechanism for fine-grained data access control with privacy conservation. The murmushash [33] mehtod is used to improve the policy privacy but with higher communication overhead.

Zhang et al. [34] introduced a new technique known match-then-re-encryption where prior to re-encryption, a matching phase was introduced. In addition, they anonumously checked whether or not the proxy could achieve a proxy re-encryption using separate re-encryption key and ciphertext components. This method was implemented using CP-ABPRE scheme based on Pairing-Based Crypto Library.

Zhang et al. [35] introduced a Privacy-Aware S-Health (PASH) access control system, in which they used CP-ABE scheme with partially hiding the access policy. In this method, they have hidden the attribute value of access policies in SHR. The performance of this

method is better compared to other methods in terms of access policy complexity, enciphering and deciphering time. Only problem here is they hidden the partial information.

Chen et al. [36] proposed a new scheme named as match-then-decrypt in which the matching phase is introduced before decryption process. This scheme is performed by calculating a distinctive constituent in a ciphertext, the constituents were used to validate that if the secret key matches ciphertext-free hidden access policy. In terms of computation time, public key size, matching phase and decryption phase, this scheme is equated with the existing CP ABE scheme. They focus on decrpytion alone, not a phase of encryption.

The Group Attribute based access control for smart cars, big data, intelligent transportation are discussed from [37–41].

The detailed comparison of various CP-ABE schemes with each and every methods techniques, features, security functionalities, computational cost is presented in Table 1. From this debate, it appears that we will have issues with access policy privacy, user security, insider attack protection, and IoT interoperability. It is noted from this literature survey that the ciphertext size and number of pairing operations vary with regard to the number of attributes in existing ABE methods. This may reduce CP-ABE methods' efficiency. The proposed CP-ABE method is novel in terms of

1. Offering encryption at first level of privacy using Ciphertext policy attribute based encryption;
2. Offering policy anonymization (SHA1) at second level;
3. Verifying the data owner authentication using BLS signature methods and thereby reducing the potential for insider attack and also;
4. Solving storage overhead problems by generating constant ciphertext size along with;
5. An enhanced IoT architecture has been designed which offers secured access control.

Table 1. Related works comparison with the proposed method.

Scheme	Technique Used	Policy Hiding	Constant Ciphertext	Insider Attack	IoT Integration	Cost
[17,18]	Inner Product Encryption	Yes	No	No	No	Moderate
[19]	CP-ABE, Inner Product Encryption	Yes	Yes	No	No	Moderate
[20,21]	CP-ABE	Yes	No	No	No	High
[22–24]	CP-ABE	No	No	No	No	Moderate
[26]	CP-ABE	Yes	No	No	No	Moderate
[27]	CP-ABE	Yes	No	Yes	No	High
[28–32]	CP-ABE	Yes	No	No	No	High
[33–35]	CP-ABPRE	Yes	No	No	No	High
[36]	CP-ABE	Yes	No	No	No	Moderate
[Ours]	CP-ABE	Yes	Yes	Yes	Yes	High

3. Preliminary

We present some facts related to groups with bilinear maps that are efficiently computable. The notations of the proposed method is mentioned in Table 2.

3.1. Basic Concepts of Bilinear Map

The bilinear map is the tool based on pairing-based cryptography. To define the bilinear map, the following notations are used.

- Let G_1 and G_2 be two multiplicative cyclic symmetric groups of the prime numbers R.
- y is the generator of G_1.
- The bilinear map e is defined as e: $G_1 * G_1 \rightarrow G_2$.

The properties of bilinear map are as listed below

1. Bilinearity: $e(y_1^m, y_2^n) \rightarrow e(y_1, y_2)^{mn}$, $\forall\ y_1, y_2 \in G_1$ and $m, n \in \mathbb{Z}_R$.
2. Non-degeneracy: $e(y_1, y_2) \neq 1$.
3. Computability: The bilinear map (e) is efficiently computable.

Table 2. Notations used for the proposed system.

Notations	Physical Meaning
$\mathbb{A}$	A set of access structure.
G	Paring groups G_1, G_2.
σ	Short signature.
y_1, y_2	Generators of G_1 and G_2.
R	Large Prime Number
h	Hash function.
AP	The user access policy.
λ	Public security parameter.
ψ, γ	Random exponents.
SK	A secret key for signing.
PK	A public key for signing.
MSK	Master Secret Key
P	Input message.
AD	An adversary.
CT	Ciphertext
CHR	The challenger.
CF	The counterfeiter.
BLS	Boneh–Lynn–Shacham Signature
SHA1	Secure Hash Algorithm
KGC	Key Generation Centre
ABE	Attribute Based Encryption
IPE	Inner Product Encryption

3.2. Access Structure

Definition of Attribute Access Structure: Considering $A_1, A_2, \ldots, A_n$ as set of user attributes, the collection $\mathbb{A} \subseteq 2^{A_1, A_2, \ldots A_n}$ is monotone if $\forall$ B, C such that if $B \in \mathbb{A}$ and $B \subseteq C$, then $C \in \mathbb{A}$. An access structure is a collection $\mathbb{A}$ of non-empty subsets of $A_1, A_2, \ldots, A_n$. The sets present in $\mathbb{A}$ are known as authorized sets, otherwise unauthorized sets.

3.3. Boneh–Lynn–Shacham (BLS) Signature

The Boneh, Lynn and Shacham (BLS) [42] introduced a simple and deterministic signature scheme. The outcome of the BLS scheme is often referred as short signatures. Although multiple signatures are adopted for cloud computing, it is not possible to verify the authentication of the data owner. We adopt BLS signature scheme for this proposed method to solve this issue. The main task of this BLS scheme is to verify the authenticity of the signer as valid or not. The short signature scheme has three different functions.

1. Key Generation: Randomly choose a number x from the interval of 0 to $R - 1$. The output of this function is the private key (x) and public verification key ($k = y^x$).
2. Signing: Given a message (m) and output of the key generation function, calculate $\sigma = h^x$ (i.e., $h = H(m)$).
3. Verification: Given a message signature (h, σ) and the public key (k), verify that (σ, y, h) is valid or not [i.e., $k = y^x$].

3.4. CP-ABE Definition

An encryption scheme based on a ciphertext-policy attribute consists of four sub basic algorithms: setup, encryption, keygen, and decryption.

1. CP-ABE-Setup (1^{λ}): It produces a public key (PK) and a master secret key (MSK) for the given security parameter λ.
2. CP-ABE-Keygen (PK, MSK, S): It produces secret key (SKS) corresponding to a set of user attributes for the given public key (PK), master secret key (MSK).
3. CP-ABE-Encryption (PK, m, $\mathbb{A}$): For given public key (PK), message (m), access structure ($\mathbb{A}$), it produces ciphertext, c.
4. CP-ABE-Decryption (PK, SK_S, c): It produces by the original message m for the given public key (PK), secret key (SKS) and ciphertext (c).

3.5. Threat Model and Goals

The proposed model addresses threats that are raised in following two levels.

1. Data Threat Level: It defines an entity which can perform data operations without proper data owner authentication, risking the confidentiality and privacy of the data received by a user.
2. Data and Access Policy Integrity Threat Level: It defines a malicious user/insider whose intent is to access the data and abuse the access rights while altering a data owner's access policy.

We aim to achieve following security objectives in order to to make the system to be resistant against above mentioned threats

1. Fine-Grained Access Control:
   Access policy of the proposed system is embedded in ciphertext to deliver access control. The access policy could be defined depending on user attributes. No one can easily change or recreate the key which is often used for offering an access control because the CP-ABE scheme is implemented from bilinear pairing.
2. Data Confidentiality:
   The user can produce the public/secret key pairs based on bilinear pairing. Hence, only the authorized user gets information about the outsourced document. Therefore, no one can violate data confidentiality.
3. Authenticity and Integrity:
   BLS verification verifies the authentication and integrity of the data outsourced by the data owner.
4. Privacy Preserving:
   In our system, the encryption method offers first level of privacy and policy anonymization scheme (SHA1) offers second level. Hence, users will not know about other users making the proposed method as fully privacy preserving.

4. Proposed System Design

Figure 3 demonstrates the architecture of the proposed hidden access control scheme that guarantees to deliver fine-grained access control along with security against insiders attack using BLS signature. The proposed system consists of four different entities.

1. Data Owner: In fact, it is the data owner's responsibility to encrypt all data using access policy before outsourcing to the cloud. The data owner also uses a hashing algorithm to hide the access policy and submit it together with the ciphertext.
2. Cloud Server: The task of a cloud server is to store the data owner's files as well as to allow the licensed users to access data. In a real-world scenario, a cloud server is honest but curious so we should hide the access policy from the cloud server.
3. User: The key generation center is responsible for creating a secret key for an individual cloud user. Only the legitimate user whose secret key satisfies the access policy can decrypt the data.
4. Key Generation Centre (KGC): The key generation center generates and distributes secret key to legitimate cloud user.

As shown in Figure 3, initially the KGC generates a public key (PK) and a master secret key (MSK) in step 1. In step 2, KGC sends a public key (PK) to the data owner. The

data owner encrypts their data at step 3 and uses SHA1 to anonymize the access policy. Data owner outsources encrypted data to the cloud server at step 4 along with anonymous access policy. In step 5, an user sends a request for data to the cloud server. The cloud server sends a ciphertext (CT) to data user in step 6. After that, the data user request a secret key for received ciphertext (CT) at step 7. In step 8, the KGC responds to the data user with a secret key. Finally, the data user decrypts ciphertext in step 9 and checks whether the signature is authentic or not.

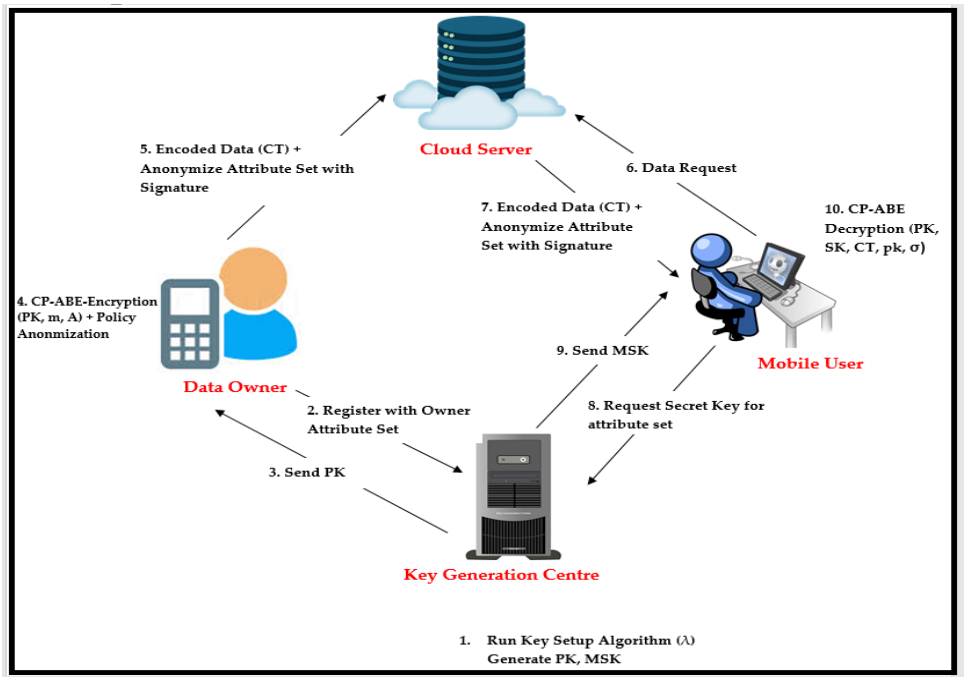

Figure 3. A system model of our proposed CP-ABE scheme.

5. Process of Proposed Scheme

CP-ABE is one of the ABE most functional and effective version. The important feature of this scheme ensures security and fine-grained access control of outsourced data. However, in an earlier version of CP-ABE, together with the ciphertext, the access policy is provided as plaintext. This may reveal the attributes of user and lead to loss of user's privacy. In the existing CP-ABE scheme, the authentication of the data owner and the integrity of outsourced data cannot be verified. To do this, we are introducing a new idea to check the data owner's authentication and check the integrity of outsourced data through BLS short signature scheme. The proposed scheme offers protection against the insider data theft.

In order to improve access policy privacy, we applied policy anonymization scheme in the proposed system. For policy anonymization, as described in Algorithm 1, we used SHA1 hashing algorithm.

Algorithm 1: Anonymization of the Access Policy

input : Access Policy (AP)
output: Anonymized Access Policy

Anon(AP)
`Function(`**Anonymization(AP)**`)` Parse the access policy, AP.
Extract attributes such that the attributes are not in the {relational operators, AND, +, =, OR, threshold gates}.
for *each attribute,* **do**
  attribute = h(attribute)
end

Throughout an ABE scheme, all insiders are considered legitimate users to obtain the original message. Therefore, it may be possible to encode the generated plaintext again using a similar or dissimilar policy of access. The short signature method is used to identify the insider theft in order to avoid this situation. The two important tasks of the short signature method are to check data owner authentication and validate the reliability of shared data. Under an adaptive message attack, this method is proven secure.

1 Setup Algorithm

- Setup(λ) $\rightarrow$ (PK, MSK)

It is the responsibility of the key generation center to run this algorithm. The KGC selects two finite prime order R random cyclic symmetric groups G_1 and G_2 with y generator. Considering λ as a public security parameter, the bilinear map is defined as e: $G_1 * G_1 \rightarrow G_2$. Randomly, KGC chooses two exponents namely ψ, $\gamma \in \mathbb{Z}_R$.

The public key PK and a master secret key MSK are generated on the basis of security parameter (λ) and two exponents (ψ, $\gamma \in \mathbb{Z}_R$).

$$\text{Public key, PK} = (G_1, y, h = y^{\gamma}, f = y^{1/\gamma}, e(y, y)^{\psi});$$
$$\text{Master Secret Key MSK} = (\gamma, y^{\psi})$$

2 Key Generation

- KeyGen (PK, MSK, A) $\rightarrow$ SK

This procedure runs the public key (PK), master secret key (MSK) and set of attributes (SA) as input, producing a secret key for the legitimate user (U_t). Based on two different random numbers M_t, $M_j \in \mathbb{Z}_R$ the secret key is generated.

$$\text{Secret Key } SK_{U_t} = (D = y(\psi + M_t)/\ \gamma, \forall \text{ in } j \in A: D_j = y^{M_t} * H(j)^{M_j}, D'(j) = y^{M_j})$$

3 Signing the Keygen

- SignKey (sk) $\rightarrow$ pk

Data owner randomly chooses a number x from the interval, 0 to R $-$ 1. The output of this function is the private key (x) and public verification key ($k = y^x$), where y is the generator of G_1

4 Encryption and Signing

- EncipherSign (PK, P, AP) $\rightarrow \sigma$, CT

The access policy is inserted into the ciphertext in our proposed system to provide access control. The access policy is expressed as an access structure as in Figure 4. The threshold gates are defined in interior nodes and user credentials/attributes are defined in leaf nodes. The access policy has been anonymized by utilizing Algorithm 1. Before starting the process, the message P is encrypted that use the public key. The proposed encryption and signature scheme is explained in Algorithm 2 with collection of leaf nodes (L).

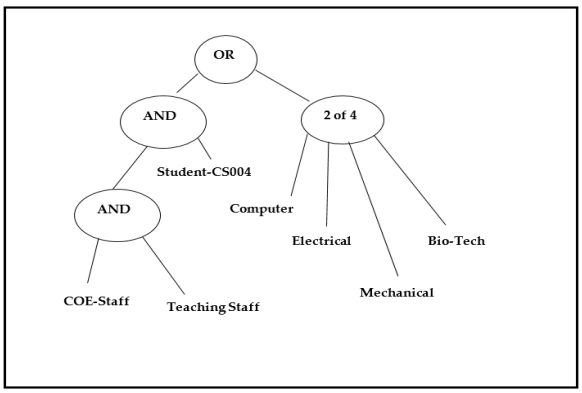

Figure 4. A sample access structure.

Algorithm 2: Algorithm for data encryption

```
Function(EncipherSign (PK, P, AP))
Anonymization (AP)
if node = = root then
    for root node W, do
        set q_W(0) = A
    end
end
C′= P · e(y, y)^{ψA};
C = h^A
if node = = leaf then
    for all leaf nodes l ∈ L do
        C_l = y^{q_l(0)}; C′_l = H(att(l))^{q_l(0)}
    end
end
Signing (P, x)
```

The output of this function can be expressed as

$$\text{Ciphertext, CT} = [\text{Anon (AP)}, \text{Sign} = \text{h (P)}^{x}, C' = P \cdot e(y, y)^{\psi A}; C = h^{A}, [C_l = y^{q_l(0)}; C'_l = H(att(l))^{q_l(0)}, \forall\, l \in L\ ]$$

The signature signing is done by BLS [15] signature scheme. Let y be the generator of gap group G_1 with the finite prime order of R and a hash function (h). Using the hash function, the short signature can be calculated and attached with the shared ciphertext (Algorithm 3).

- Signing(P, x) $\rightarrow \sigma$

where σ = = h (P^x).

Algorithm 3: Algorithm to generate a signature

```
Function(Function Signing (P, x))
Compute hash(P)
Assign signature σ = h(P^x)
```

Figure 5 illustrates the outsourced file structure in the cloud. The first column represents the unique identity of shared data, the second column represents the signature (P) and final column represents the ciphertext (CT).

UUID	Signature(Sig)	Encrypted File with Hidden Access policy (CT)

Figure 5. The outsourced data structure in the cloud.

5 Decryption and Verification

- DecipherVerify (PK, SK_{u_t}, CT, σ, pk) $\rightarrow$ P, Success/Failure

The decryption operation is successful whenever the attributes of access policy, which is embedded inside the ciphertext is matched with the attribute of the cloud user. If it is not, then the cloud user cannot decrypt the ciphertext. The BLS short signature is used to check whether the data owner is authentic or not, in order to avoid the insider attack. The decryption process of our proposed method is shown in Algorithms 4 and 5.

Algorithm 4: Algorithm for Decryption and Verification

```
Function(DecipherVerify (PK, SK_{u_t}, CT, σ, pk))
DecipherNode (CT, SK, l)
if policy is satisfied by A then
    A = DecipherNode(CT, SK, P) = e (y, y)^{MA}
    C′ = P · e(y, y)^{ψA}; e (C, D) = e (y^{ψA}, y^{(ψ+M/γ)})
    P = C′ / (e (C, D) / A)
end
Verify(σ, pk)
```

Algorithm 5: Algorithm for Node Decryption

```
Function(DecipherNode (CT, SK, l))
for each leaf node l do
    assign j = attr(l)
    if j ∈ A then
        DecipherNode = e (D_j, C_x) / e (D′_j, C′_x)
        return (e (y, y)^{Mq_l(0))}
    end
    else
        return null
    end
end
```

Suppose leaf node l is not in L, access structure (SA) is satisfied by access tree. Now, decipher function will repeatedly compute and return e (y, y)$^{Mq_l(0))}$ = e(y, y)^{MA}. After that, the plaintext (P) can be easily calculated from the ciphertext (CT).

6 Signature Verification

This algorithm takes a message (P), computed hash (σ) and the public key of users (pk) and verifies the signature as shown in Algorithm 6

$$\text{Verify (P, } \sigma\text{, pk)} = \text{e (}\sigma\text{, y)} = \text{e (h (P), y}^x\text{)}$$
$$\text{e (h (P}^x\text{), y)} = \text{e (h(P), y}^x\text{)}$$

Algorithm 6: Algorithm to Verify the Signature

```
Function(Verify (P, σ, pk))
if e (σ, y) = e (h(P), y^x) then
    return Success
    else
        return Failure
    end
end
```

5.1. Correctness Proof of Our Proposed Method

From Algorithm 4, e (D, C) can be written as,

$$\begin{aligned} \text{e (D, C)} &= \text{e}(\text{y}^{(\psi+M_t)/\gamma}, \text{y}^{\gamma A}) \\ &= \text{e (y, y)}^{(\psi+M)A} \\ \text{e (D, C)} &= \text{e (y, y)}^{\psi A} \cdot \text{e (y, y)}^{MA} \end{aligned}$$

substitute e (D, C) value in Algorithm 5.

$$\begin{aligned} \text{e (D, C)/A} &= \text{e (y, y)}^{\psi A} \cdot \text{e (y, y)}^{MA} / \text{e (y, y)}^{MA} \\ &= \text{e (g, g)}^{\psi A} \end{aligned}$$

Finally, substitute the value of e (D, C)/A in Algorithm 4.

$$C'/e\,(D, C)/A = P \cdot e\,(y, y)^{\psi A}/e\,(y, y)^{\psi A}$$

5.2. Application Scenario for Proposed Method

Figure 6 illustrates the scenario of Hospital Information System when data is shared between users (Patients, Insurance Companies, Government bodies etc.) and Smart Devices (Data Owners). The example scenario work flow gets started by key generation authorities to generate the Public key and Master Secret Key (MSK). Then, each smart device registers with key generation authorities by sending their attributes. After registration, KGC sends Public Key (PK) to the concerned smart device. With this key, the smart device can encrypt its data using Ciphertext Policy Attribute Based Encryption and also can hide the access policy using policy anonymization method. To verify the authenticity of smart devices, short signature scheme (BLS) is used. The encoded files, anonymized policy, signature is uploaded to the hospital service provider (HIS). At that point, the patients or smart devices send requests to HIS and in turn, HIS sends encrypted data, anonymized policy, and signature to the patients or smart devices. In order to view the encoded data, patients or smart devices need to complete the decoding process. During decoding, the order is matching of policy, matching of the data owner attributes and finally verifying the data owner signature. If anyone of these three matching is not satisfied, the opponents or users will not be able to view the original information of the data owner.

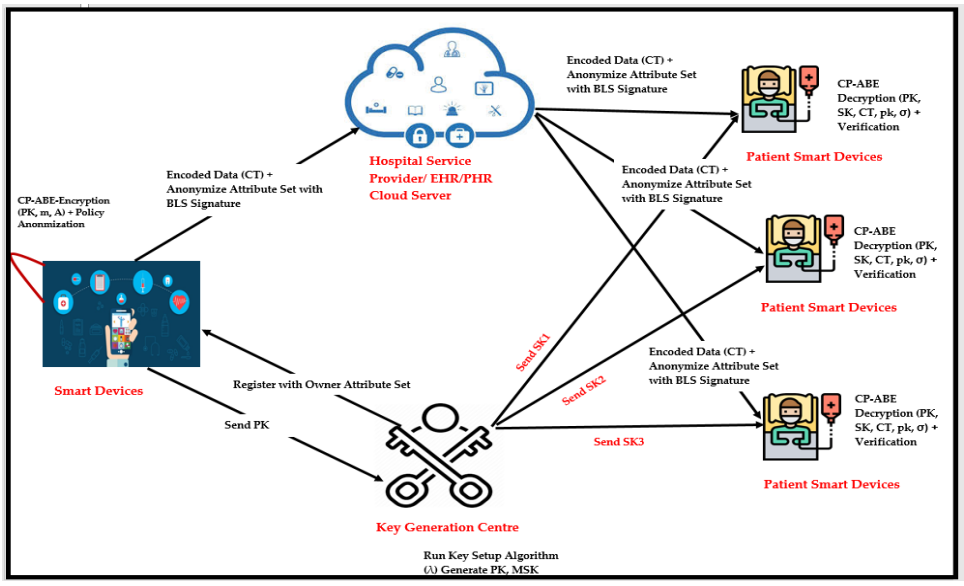

Figure 6. An application scenario for the proposed method.

6. Implementations and Performance Evaluation

All the operations of the proposed technique have been experimented on Intel Core i5-4440 CPU @ 3.10GHz processor with 8GB RAM running on Microsoft Windows-10 64-bit operating systems. We also used an Android 7.0.1 Honor mobile with Octa-core processor and 3GB RAM as the IoT device. Typically, the java based CP-ABE toolkit [43,44] using jPBC library (version 2.0.0) [45] is utilized to implement the proposed system. In the access policy, the number of user attributes varies from 10 to 100 [46].

6.1. Performance Evaluation

In terms of encryption, decryption and memory analysis, we evaluate the performance of the proposed method and compare it other existing CP-ABE scheme such as [35,47,48].

6.1.1. Time Comparison of Proposed Method with CP-ABE

The proposed technique uses policy anonymization, improves the privacy policy and signature verification of the data owner and identifies the insider attack. In order to achieve this policy anonymization, Secure Hashing Algorithm (SHA1) was used. However, this hashing method introduces a negligible overhead at the data user part.

As shown in Figure 7, existing CP-ABE methods with policy hiding [26,27,47,48] consumed 0.078, 0.18, 0.21 and 0.18 s to generate a key of 10 attributes, whereas the proposed method took only 0.1 s. Similarly, for 100 attributes, the key generation times were 0.545, 0.792, 0.82 and 0.795 s, respectively. The proposed method, however, took 0.57 s, which is lesser compared to the existing methods is shown in Table 3. Further, the proposed method also hides the access policy.

Table 3. Execution time of key generation centre (KGC).

Number of Attributes	Helil et al. [26]	Sabitha et al. [27]	Odelu et al. [48]	Wu et al. [47]	Proposed Method
10	0.078	0.18	0.21	0.18	0.1
20	0.107	0.2	0.25	0.21	0.13
30	0.16	0.28	0.31	0.28	0.2
40	0.2	0.32	0.38	0.31	0.245
50	0.28	0.38	0.44	0.42	0.314
60	0.385	0.45	0.52	0.49	0.398
70	0.402	0.56	0.61	0.56	0.418
80	0.478	0.61	0.69	0.63	0.49
90	0.512	0.68	0.76	0.71	0.525
100	0.545	0.792	0.82	0.795	0.57

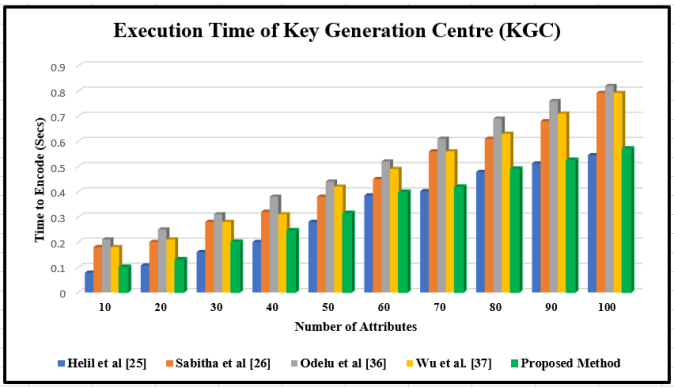

Figure 7. Execution time of the key generation centre (KGC).

Figure 8 shows the results of the comparison of encryption time from the data owner side. The existing methods took 0.34, 0.372, 0.384 and 0.393 s for encryption with 10 attributes, whereas the proposed method consumed only 0.361 s. Similarly, for 100 attributes, the encryption times were 1.532, 1.65, 1.73, and 1.72 s, respectively, is shown in Table 4. The proposed method, however, took 1.541 s, which is lesser compared to the existing methods. Further, the proposed method also hid the access policy. Hence, the proposed method has been proved to have provided better security and privacy compared to existing methods.

As shown in Figure 9, existing CP-ABE methods [26,27,47,48] took 0.015, 0.021, 0.025, and 0.028 s to decipher 10 attributes, whereas the proposed method took only 0.031 s. Similarly, for 100 attributes, the decryption time were 0.054, 0.0821, 0.083 and 0.083 s, respectively. However, the proposed method took 0.114 s, which is higher than the existing methods due to the adaptation of policy anonymization is shown in Table 5. However, the proposed method increased the privacy of access policy as well as shared data by hiding the access policy. Further, the proposed method offered a unique feature to prevent this insider attack by utilizing a short signature scheme.

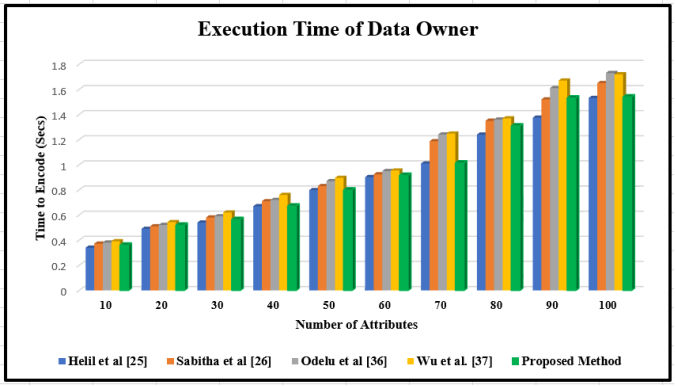

Figure 8. Encipher time in data owner part.

Table 4. Execution time of the data owner.

Number of Attributes	Helil et al. [26]	Sabitha et al. [27]	Odelu et al. [48]	Wu et al. [47]	Proposed Method
10	0.34	0.372	0.384	0.393	0.361
20	0.49	0.51	0.521	0.543	0.52
30	0.54	0.58	0.59	0.62	0.565
40	0.67	0.71	0.72	0.76	0.672
50	0.798	0.83	0.87	0.896	0.8
60	0.902	0.924	0.95	0.954	0.917
70	1.01	1.187	1.241	1.248	1.015
80	1.24	1.35	1.36	1.369	1.31
90	1.375	1.52	1.61	1.67	1.532
100	1.532	1.65	1.73	1.72	1.541

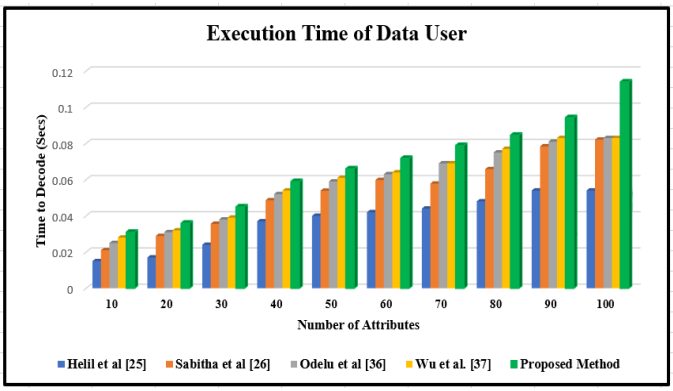

Figure 9. Decipher time in user part.

Table 5. Execution time of the data user.

Number of Attributes	Helil et al. [26]	Sabitha et al. [27]	Odelu et al. [48]	Wu et al. [47]	Proposed Method
10	0.015	0.021	0.025	0.028	0.031
20	0.017	0.0289	0.031	0.032	0.036
30	0.024	0.0356	0.038	0.039	0.045
40	0.037	0.0486	0.052	0.054	0.059
50	0.04	0.0539	0.059	0.061	0.066
60	0.042	0.0598	0.063	0.064	0.0718
70	0.044	0.0578	0.069	0.069	0.0789
80	0.048	0.0658	0.075	0.077	0.0846
90	0.054	0.0784	0.081	0.083	0.0943
100	0.054	0.0821	0.083	0.083	0.114

6.1.2. Overhead

The overhead of the proposed method is measured at both owner and cloud server part. Figures 10 and 11 showed that the proposed method occupy the constant memory to store the secret key and encrypted file. However, in [27], both secret key and encrypted file size increase linearly with respect to number of user attributes. Therefore, considering the number of secret key attributes (e.g., a value is three), the length of the secret key will increase quadratically with the number of N attributes as shown in Figure 10.

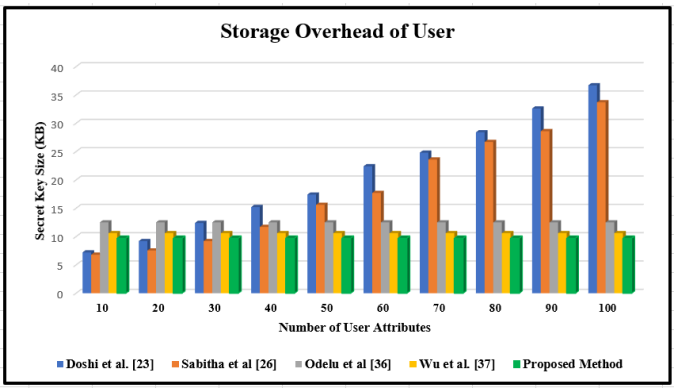

Figure 10. Secret key size at the user's side.

The ciphertext stored in the cloud consists of the UUID, the signature, and the hidden access policy ciphertext of data. The EncryptSign algorithm's result includes policy anonymization, signature, and ciphertext, so it had three attributes entirely. The Table in Section 6.3 shows that the ciphertext size increases with respect to the number of attributes in existing schemes. The proposed method is efficient in terms of memory consumption, security and time analysis compared to the existing scheme.

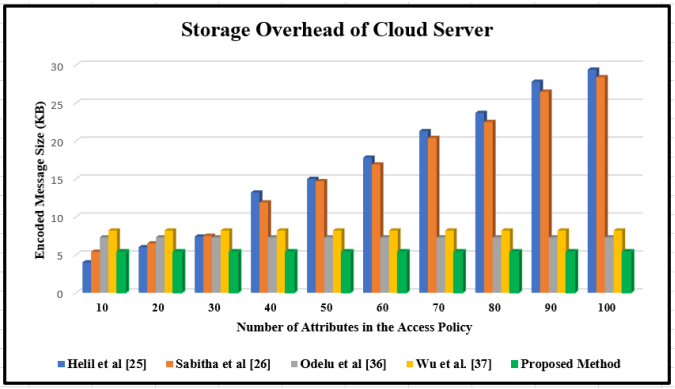

Figure 11. Encrypted file size in the cloud server.

6.2. AI enabled IoT Usecase Performance Evaluation

The current study utilized the core C implementation of CP-ABE proposed in [49] in order to implement this use case. The C code is accessed on 24 February 2020 at http://spritz.math.unipd.it/projects/andraben. The number of records, throughout the dataset [49], lies in the range of 10 to 50. The tests are carried out for 10 times to ensure that the findings are accurate and consistent enough to compare it with the existing methods. The proposed study was contrasted against the scheme developed by Odelu et al. [48] and Dmitrienko et al. [49] in terms of key generation time and encoding and decoding time.

6.2.1. Key Generation Time

As shown in Figure 12, the existing method by Odelu et al. [48] and Dmitrienko et al. [49] took 5.5 and 5.7 s to generate a key of 10 records, whereas the proposed method

consumed 6.1 s. Similarly, for 50 records, the key generation time was 7.8 and 8.2 s for the existing method. However, the proposed method took 8.8 s, which is, to some extent, higher than the existing method is clearly mentioned in Table 6. This additional time is due to the adaptation of the policy anonymization technique to enhance the privacy of the user.

Table 6. Comparison of key generation time in IoT device.

Number of Attributes	Odelu et al. [48]	Dmitrienko et al. [49]	Proposed Method
10	5.5	5.7	6.1
20	5.7	5.9	6.7
30	6.2	6.8	7.5
40	7.1	7.4	7.9
50	7.8	8.2	8.8

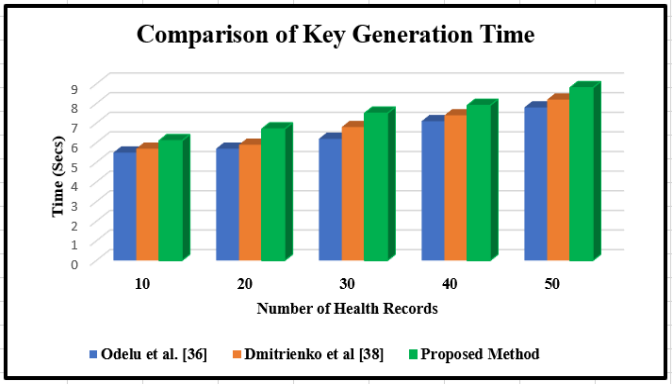

Figure 12. Comparison of key generation time in IoT environment.

6.2.2. Encoding Time

Figure 13 shows the encoding comparison of IoT-based architecture. The existing method by Odelu et al. [48] and Dmitrienko et al. [49] took 0.16 and 0.2 s to encode 10 records, whereas the proposed method consumed 0.31 s. Similarly, for 50 records, the encoding time was 0.88 and 0.931 s for the existing method. However, the proposed method consumed 1.1 s. The additional time, consumed by the proposed method, improves the confidentiality of user data by hiding both access policy as well as CP-ABE encryption is shown in Table 7.

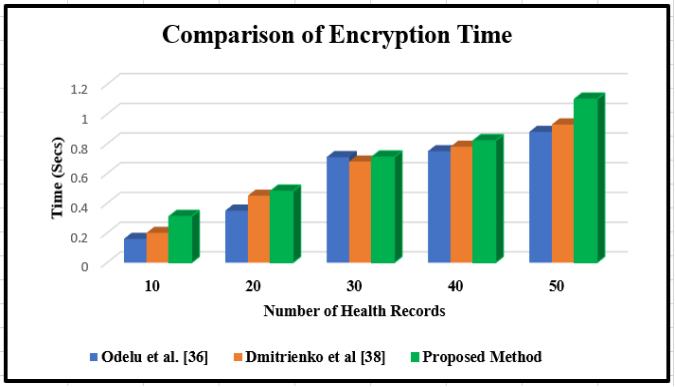

Figure 13. The encoding time in IoT environment.

Table 7. Comparison of encoding time in IoT device.

Number of Attributes	Odelu et al. [48]	Dmitrienko et al. [49]	Proposed Method
10	0.16	0.2	0.31
20	0.35	0.45	0.48
30	0.71	0.68	0.71
40	0.75	0.78	0.82
50	0.88	0.93	1.1

6.2.3. Decoding Time

Figure 14 shows the decoding time on the user side. The existing method of Odelu et al. [48] and Dmitrienko et al. [49] consumed 0.28 and 0.36 s to decode 10 records, whereas the proposed method took 0.52 s. Similarly, for 50 records, the decoding time was 0.984 and 1.25 s in the case of the existing method. The proposed method, however, consumed 1.54 s. The additional time required by the proposed method is due to policy anonymization and BLS scheme. Moreover, the security of the proposed method is high compared to the existing methods since three important security measures have been incorporated in the proposed method, such as data owner verification, policy anonymization, and access control. The Table 8 shown the detailed comparison of existing methods and proposed method.

Table 8. Comparison of decoding time in IoT device.

Number of Attributes	Odelu et al. [48]	Dmitrienko et al. [49]	Proposed Method
10	0.28	0.36	0.52
20	0.398	0.58	0.68
30	0.546	0.75	0.88
40	0.764	0.84	0.93
50	0.9	1.25	1.54

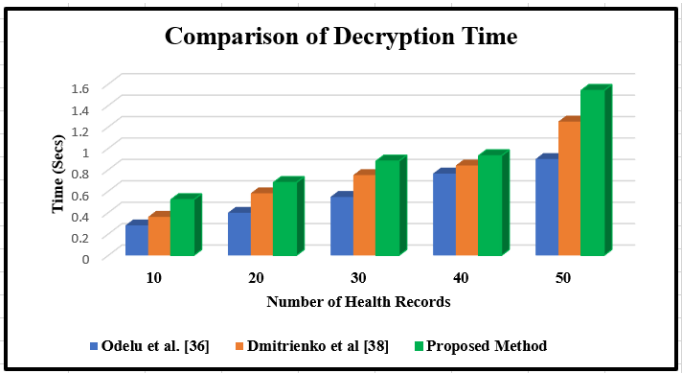

Figure 14. The decoding time in IoT environment.

6.2.4. Attack Analysis on Proposed Method

Figure 15 shows the time taken by an intruder to collapse the existing as well as the proposed system. For this validation, one of the patient roles was considered as an intruder, as discussed under Section 7.3. The existing method, by Odelu et al. [48] and Dmitrienko et al. [49], got compromised in 0.68 and 1.25 s to crack a case of 10 attributes, whereas the proposed method consumed 1.78 s to get compromised. When the attributes increase, the resistance time of the proposed method becomes highly notable. Considering 50 attributes, the interrupt time was 2.67 and 4.27 s for the existing method and 6.21 s for the proposed method is shown in Table 9. This result clearly indicates that the proposed method offers an enhanced level of security, compared to the existing methods, in the IoT scenario.

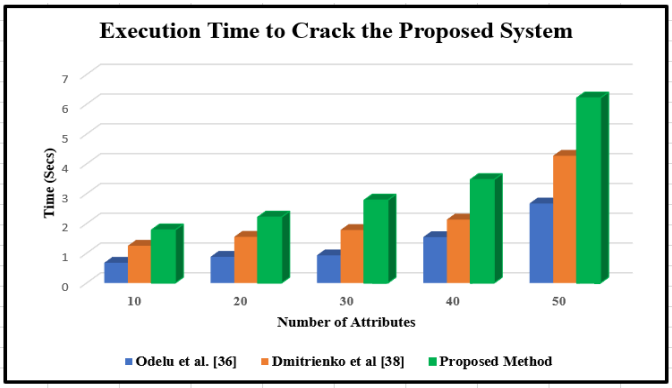

Figure 15. The execution time to crack the proposed system.

Table 9. Execution Time to crack the proposed system.

Number of Attributes	Odelu et al. [48]	Dmitrienko et al. [49]	Proposed Method
10	0.68	1.25	1.78
20	0.88	1.55	2.21
30	0.93	1.78	2.78
40	1.54	2.13	3.47
50	2.67	4.27	6.21

6.3. Comparative Analysis

From Table 10, we can see that in most of the CP-ABE schemes, the ciphertext size is not constant and hence the memory consumption still remains as a problem. Although the scheme offered a selective secure without random oracles in [14,22,26–28,47,48], it did not generate a constant ciphertext size, whereas the proposed CP-ABE scheme is selectively secure but has a constant ciphertext size without random oracles.

Table 10. Comparison among different CP-ABE schemes.

Scheme	Random Oracles	Security Model	Access Structure	Hidden Access Policy	Ciphertext Size
CP-ABE [14]	YES	Generic group Model	LSSS	NO	O(n)
CP-ABE [22]	NO	Selective	LSSS	No	O(n)
CP-ABE + Hidden Access policy[26]	NO	Selective	AND Gates	YES	O(n)
CP-ABE + Hidden Access policy [27]	NO	Selective	AND gates	YES	O(n) + G_T
CP-ABE + Partially Hidden Access policy [28]	NO	Selective	LSSS	YES	O(n)
CP-ABE + Hidden Access policy [48]	NO	Selective	AND gates	YES	$O(n^2) + G_T$
CP-ABE + Hidden Access policy [47]	NO	Selective	LSSS gates	YES	O(n) + G_T
Proposed Method	NO	Selective	AND Gates	YES	O(1)

Here, n refers the number of attributes, G_T refers the prime order groups.

6.4. Computation Cost Analysis

We can see from Table 11 that the proposed scheme takes slightly longer to encrypt and decrypt than other strategies for CP ABE with hidden access policies in [26,27,47,48], because there are only three or four multiplications activities in encrypting and two or one bilinear pairing procedures in decoding. The amount of attributes in the access policy has no bearing on the exponential and bilinear pairing processes. However, our solution is more secure than the others, especially in terms of preventing insider assaults.

Table 11. Comparison of computations cost analysis with different CP-ABE schemes.

Scheme	Encoding Cost **	Decoding Cost **
CP-ABE + Hidden Access policy [26]	$3T_e$	$3T_b$
CP-ABE + Hidden Access policy [27]	$4T_e$	$3T_b$
CP-ABE + Partially Hidden Access policy [28]	$4T_e$	$3T_b + T_e$
CP-ABE + Hidden Access policy [48]	$3T_e$	$3T_b + T_e$
CP-ABE + Hidden Access policy [47]	$3T_e$	$2T_b + T_e$
Proposed Method	$3T_e$	$2T_b + 2T_e$

** where T_e represents the time for one exponential operations, T_b represents the time for one bilinear operations.

7. Security Evaluation

We analyze the impact of indistinguishable adaptive chosen-ciphertext attacks for our proposed method using a two-player game.

7.1. Indistinguishable Adaptive Chosen-Ciphertext Attack (IND-CCA2)

The security model of the proposed CP-ABE method against an indistinguishable adaptive chosen ciphertext attack (IND-CCA2) is explained by an activity between an adversary (AD) and a challenger (CHR), as below:

1. Init: The dare access structure, CAS*, is declared by an adversary (AD).
2. Setup: A challenger (CHR) selects a public security parameter, λ, processes the setup algorithm, distributes a public key (PK) to the adversary (AD), and secretly saves the master secret key (MSK). Two random exponents, ψ, $\gamma \in \mathbb{Z}_R$ are selected.
3. Phase 1: The adversary (AD) creates a polynomial time request and the challenger (CHR) provides consistent answers.
   (a) Secret Key Request: The adversary (AD) creates a continuous request to produce a secret key, SK_{u_t}, with a set of user attributes, W_1, W_2,…,$W_n \in A$. The challenger (CHR) replies with the secret key, SK_{u_t}, by running the KeyGen algorithm. A number, r_j, is randomly chosen and matched to every attribute of $j \in A$. The secret key can be calculated as $D_j = y^{r_j} * H\ W\ (j)^{r_j}$, $D'\ (j) = y^{r_j}$ and distributed to the adversary (AD).
   (b) Decipher Request: The adversary (AD) selects a ciphertext randomly decrypted by the access structure (CAS*). The challenger (CHR) then runs the DecipherVerify algorithm to decipher the ciphertext using the SK_{u_t} The resultant plaintext is sent to the adversary (AD).
4. Challenge: The adversary (AD) submits two messages of equal length, P_0, P_1, to the challenger (CHR). The CHR chooses a random number, $\alpha \in 0, 1$, and encrypts the message, P_α, using the challenge access structure, CAS*. The resultant ciphertext (CT*) is sent to the AD.
5. Phase 2: Similar to Phase 1, the restriction here is that the submitted ciphertext, CT, is not identical to the generated ciphertext, CT*.
6. Guess: The adversary (AD) outputs a guess, α' of α.

At the end of this game, we say that an adversary has succeeded if $\alpha' == \alpha$. The advantage of this game is defined as

$$Adv_{IND-CCA2} = Pr\ [\alpha' = = \alpha] - \frac{1}{2}.$$

From this two-player game, we conclude that the proposed system offers security against an indistinguishable adaptive chosen-ciphertext attack (IND-CCA2).

7.2. Security against Secure Signatures

Definition 1 (Security of Signatures). *A counterfeiter (CF) [50] may be expected to (t, Q_H, Q_{sig}, ϕ) break the BLS scheme σ { Keygen, Sign, Verify} using an adaptive chosen message attack if, after several Q_H enquiries to the hash model with t processing time, it produces an output as a valid counterfeiter with the probability of a minimum ϕ.*

A signature method σ(t, Q_H, Q_{sig}, ϕ) is said to be secure if no counterfeiter can break the (t, Q_H, Q_{sig}, ϕ) scheme.

Proof. In the short signature method, we need to make a hash query (Q_H) before signing the message, P. □

Suppose a counterfeiter (CF) (t, Q_H, Q_{sig}, ϕ) breaks the BLS method by means of an adaptive chosen message attack. This can be explained by a two-player game against an adversary (AD) and thecounterfeiter (CF).

1. Step 1: The counterfeiter (CF) runs a SignKey algorithm and produces a public key (pk) and a secret key (sk). The public key is forwarded to the adversary (AD).
2. Step 2: The adversary (AD) makes a continuous polynomial time request (t) and the counterfeiter (CF) answers hash and signature enquiries.
   (a) Hash enquiries (Q_H): The adversary makes a request on a hash oracle model on messages P_k for $1 \leq k \leq Q_H$. The counterfeiter (CF) responds with responses already prepared for hash queries ($X_1, X_2, \ldots, X_{QH}$) on P_k.
   (b) Signature Enquiries (Q_{sig}): The adversary (AD) makes a request on a signature oracle model for the given messages, $P_k = (p_1, p_2, \ldots, p_{Qsig}) \in 0, 1^*$, using the public key (pk). The counterfeiter (CF) responds with a signature ($\sigma_k \rightarrow$ Signing (P_k, sk) to the adversary (AD).
3. Step 3: Finally, the counterfeiter (CF) halts the process, gives the message signature pair as (P, σ), verifies the validity of (P, σ, pk), and outputs (P_k, σ, X_1, $X_2, \ldots, X_{QH}$). The counterfeiter, however, will fail our BLS method.

It is evident from the two-player game that any advantage an adversary has is small. Therefore, the proposed CP-ABE method has shown itself secure against the adaptive chosen message attack, since no counterfeiter (CF) (t, Q_H, Q_{sig}, ϕ) can break the system for signature (Q_{sig}) enquiries. In case the counterfeiter (CF) does manage to break the system, the adversary still has a ϕ advantage.

7.3. Threat Model for AI enabled IoT Devices

7.3.1. Case 1: Mismatching of Attributes

In this scenario, the patient acts as intruder trying to access the encrypted information of medical data. The intruder may pass all the intermediate levels except attribute authorization process from KGC. However, KGC gives authorization only to the attribute matched user. Otherwise, proposed system denies the access for intruder is shown in Figure 16.

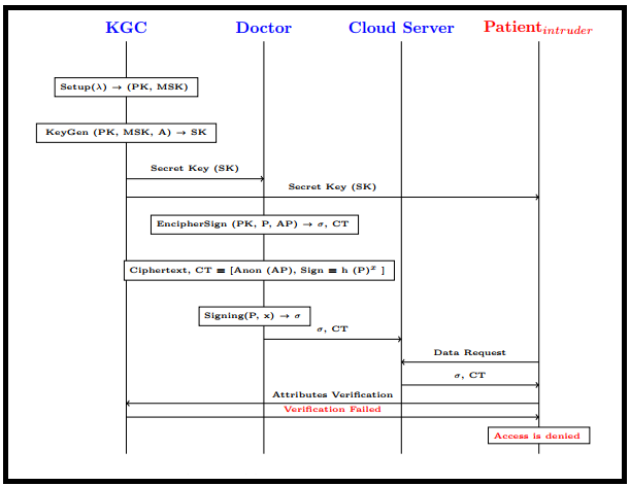

Figure 16. Threat model—mismatching of attributes.

7.3.2. Case 2: Hash Value of Access Policy Mismatching

In Figure 17, we depict the scenario where the intruder may overcome the attribute checking process but fail to match the hash value of the access policy. This is possible due to the policy anonymization method which helps to achieve two important security properties such as confidentiality and authentication.

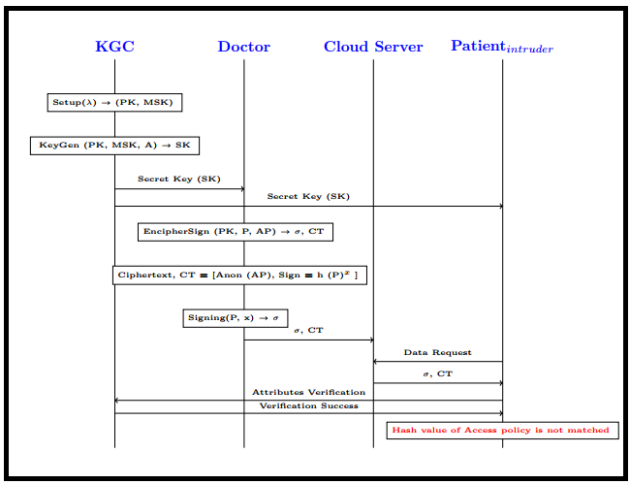

Figure 17. Threat model—hash value of access policy mismatching.

7.3.3. Case 3. Data Owner Verification

Figure 18 setup explains verifying the authenticity of the data owner or AI enabled IoT devices. Let us consider the scenario in which intruder has cracked the hash value and access policy matching of our proposed system. To handle such a case, we have further imposed additional security mechanism to verify the authenticity of the data owner using the BLS short signature scheme. This permits only the authorized user to run the verification function of BLS scheme and hence ensures privacy of the data owner.

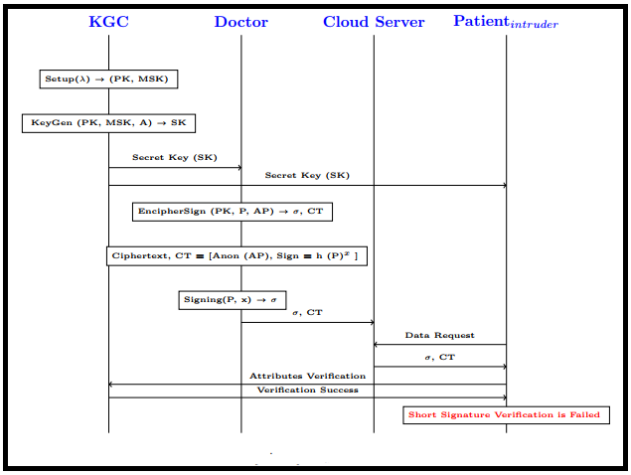

Figure 18. Threat model—policy anonymization failed.

8. Conclusions

The encryption based on attributes provides security for outsourced data as well as fine-grained control of access. In addition, the proposed scheme provides privacy protection for outsourced cloud data and verifies the integrity of shared data and insider attack security. The proposed scheme prevents masquerading, repudiation, and shows additional privacy preserving features by policy anonymization method. The efficiency of our method is better compared to existing policy-hidden methods because of constant ciphertext size. Then, we proved the proposed scheme's security against chosen ciphertext attack and insider attack using secured signature. In the future, we plan to apply this proposed method with the multi-authorities attribute set in the small healthcare sector.

Author Contributions: Conceptualization, P.C.; Data curation, P.C.; Formal analysis, P.C. and A.K.D.; Funding acquisition, J.Y.; Investigation, P.D. and G.P.J.; Methodology, P.D. and A.K.D.; Project administration, J.Y. and G.P.J.; Resources, J.Y. and G.P.J.; Software, P.D.; Supervision, J.Y. and G.P.J.; Visualization, G.P.J.; Writing—original draft, P.C.; Writing—review & editing, G.P.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No. 2020-0-00107, Development of the technology to automate the recommendations for big data analytic models that define data characteristics and problems).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Li, X.; Wang, Q.; Lan, X.; Chen, X.; Zhang, N.; Chen, D. Enhancing Cloud-Based IoT Security through Trustworthy Cloud Service: An Integration of Security and Reputation Approach. *IEEE Access* **2019**, *7*, 9368–9383. [CrossRef]
2. Joshi, G.P.; Acharya, S.; Kim, C.S.; Kim, B.S.; Kim, S.W. Smart solutions in elderly care facilities with RFID system and its integration with wireless sensor networks. *Int. J. Distrib. Sens. Netw.* **2014**, *7*, 713946. [CrossRef]
3. Riad, K.; Hamza, R.; Yan, H. Sensitive and Energetic IoT Access Control for Managing Cloud Electronic Health Records. *IEEE Access* **2019**, *7*, 86384–86393. [CrossRef]
4. Buyya, R.; Vecchiola, C.; Selvi, S.T. *Mastering Cloud Computing: Foundations and Applications Programming*, 1st ed.; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 2013.
5. Jansen, W.A.; Grance, T. *Guidelines on Security and Privacy in Public Cloud Computing*; National Institute of Standard and Techology, U.S. Department of Commerce: Washington, DC, USA, 2011.

6. Kallahalla, M.; Riedel, E.; Swaminathan, R.; Wang, Q.; Fu, K. Plutus: scalable secure file sharing on untrusted storage. In Proceedings of the USENIX Conference on File and Storage Technologies (FAST), San Francisco, CA, USA, 31 March–2 April 2003; pp. 29–42.
7. Anderson, R. *Security Engineering: A Guide to Buliding Dependable Distributed Systems*; John Wiley & Sons: Hoboken, NJ, USA, 2001.
8. Bell, D.E. Looking back at the Bell-La Padula model. In Proceedings of the 21st Annual Computer Security Applications Conference (ACSAC'05), Tucson, AZ, USA, 5–9 December 2005; pp. 1–15.
9. Wang, G.; Liu, Q.; Wu, J. Achieving fine-grained access control for secure sharing on cloud servers. *Concurr. Comput. Pract. Exp.* **2011**, *23*, 1443–1464. [CrossRef]
10. Zhu, H.; Wang, L.; Ahmad, H.; Niu, X. Key-Policy Attribute-Based Encryption with Equality Test in Cloud Computing. *IEEE Access* **2017**, *5*, 20428–20439. [CrossRef]
11. Huang, X.; Susilo, W.; Mu, Y.; Zhang, F. Short designated verifier signature scheme and its identity based variant. *Int. J. Netw. Secur.* **2008**, *6*, 82–93.
12. The Boneh-Lynn-Shancham Signature. Available online: https://en.wikipedia.org/wiki/Boneh_Lynn_Shacham (accessed on 21 December 2020).
13. Sahai, A.; Waters, B. Fuzzy Identity-Based Encryption. In *Proceedings of the Annual International Conference on the Theory and Applications of Cryptographic Techniques*; Springer: Berlin, Germany, 2005; pp. 457–473.
14. Waters, B. Ciphertext-Policy Attribute-Based Encryption: An Expressive, Efficient and Provably Secure Realization. In *Public Key Cryptography—PKC 2011*; Lecture Notes in Computer Science; Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2011; Volume 6571.
15. Miltiadis, M.; Virvilis, N.; Gritzalis, D. The insider threat in cloud computing. In *International Workshop on Critical Information Infrastructures Security*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 93–103.
16. Arrtibute-Based Encryption in Wikipedia, Retrieved 24 June 2018. Available online: https://en.wikipedia.org/wiki/Attribute-based_encryption (accessed on 25 December 2020).
17. Katz, J.; Sahai, A.; Waters, B. Predicate encryption supporting disjunctions, polynomial equations, and inner products. *J. Cryptol.* **2013**, *26*, 191–224. [CrossRef]
18. Patil, P.A.; Joshi, S. Hidden CP-ABE to Enhance Patient Data Privacy in Smart Healthcare Systems. *Int. J. Appl. Eng. Res.* **2017**, *12*, 3950–3960.
19. Phuong, T.V.X.; Yang, G.; Susilo, W. Hidden Ciphertext Policy Attribute-Based Encryption Under Standard Assumptions. *IEEE Trans. Inf. Forensics Secur.* **2016**, *11*, 35–45. [CrossRef]
20. Frikken, K.; Atallah, M.; Li, J. Attribute-Based Access Control with Hidden Policies and Hidden Credentials. *IEEE Trans. Comput.* **2006**, *55*, 1259–1270. [CrossRef]
21. Yao, X.; Liu, H.; Ning, H.; Yang, L.T.; Xiang, Y. Anonymous Credential-Based Access Control Scheme for Clouds. *IEEE Cloud Comput.* **2015**, *2*, 34–43. [CrossRef]
22. 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.
23. Doshi, D.; Jinwala, D.C. Fully secure ciphertext policy attribute-based encryption with constant length ciphertext and faster decryption. *Sec. Commun. Netw.* **2014**, *7*, 1988–2002. [CrossRef]
24. Gorasia, N.; Srikanth, R.R.; Doshi, N.; Rupareliya, J. Improving Security in Multi Authority Attribute Based Encryption with Fast Decryption. *Procedia Comput. Sci.* **2016**, *79*, 632–639. [CrossRef]
25. Miao, Y.; Liu, X.; Choo, K.K.R.; Deng, R.H.; Li, J.; Li, H.; Ma, J. Privacy-Preserving Attribute-Based Keyword Search in Shared Multi-owner Setting. *IEEE Trans. Dependable Secur. Comput.* **2021**, *18*, 1080–1094. [CrossRef]
26. Helil, N.; Rahman, K. CP-ABE Access Control Scheme for Sensitive Data Set Constraint with Hidden Access Policy and Constraint Policy. *Secur. Commun. Netw.* **2017**, *2017*, 2713595. [CrossRef]
27. Sabitha, S.; Rajasree, M.S. Access control based privacy preserving secure data sharing with hidden access policies in cloud. *J. Syst. Archit.* **2017**, *75*, 50–58. [CrossRef]
28. Liu, L.; Lai, J.; Deng, R.H.; Li, Y. Ciphertext-policy attribute-based encryption with partially hidden access structure and it's application to privacy-preserving electronic medical record system in cloud environment. *Secur. Commun. Netw.* **2016**, *9*, 4897–4913. [CrossRef]
29. Balu, A.; Kuppusamy, K. Ciphertext policy Attribute based Encryption with anonymous access policy. *Int. J. Peer Peer Netw.* **2010**, *1*, 1–8. [CrossRef]
30. Yadav, U.C.; Ali, S.T. Ciphertext policy-hiding attribute-based encryption. In Proceedings of the 2015 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Kochi, India, 10–13 August 2015; pp. 2067–2071.
31. Zhong, H.; Zhu, W.; Xu, Y.; Cui, J. Multi-authority attribute-based encryption access control scheme with policy hidden for cloud storage. *Soft Comput.* **2018**, *22*, 243–251. [CrossRef]
32. Yang, K.; Han, Q.; Li, H.; Zheng, K.; Su, Z.; Shen, X. An Efficient and Fine-Grained Big Data Access Control Scheme with Privacy-Preserving Policy. *IEEE Internet Things J.* **2017**, *4*, 563–571. [CrossRef]
33. Austin Appleby, MurmurHash. 2011. Available online: https://sites.google.com/site/murmurhash/ (accessed on 27 October 2020).

34. Zhang, Y.; Li, J.; Chen, X.; Li, H. Anonymous attribute-based proxy re-encryption for access control in cloud computing. *Secur. Commun. Netw.* **2016**, *9*, 2397–2411. [CrossRef]
35. Zhang, Y.; Zheng, D.; Deng, R.H. Security and Privacy in Smart Health: Efficient Policy-Hiding Attribute-Based 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. Gupta, M.; Benson, J.; Patwa, F.; Sandhu, R. Dynamic Groups and Attribute-Based Access Control for Next-Generation Smart Cars. In Proceedings of the Ninth ACM Conference on Data and Application Security and Privacy, Richardson, TX, USA, 25–27 March 2019; pp. 61–72. [CrossRef]
38. Gupta, M.; Patwa, F.; Sandhu, R. An Attribute-Based Access Control Model for Secure Big Data Processing in Hadoop Ecosystem. In Proceedings of the Third ACM Workshop on Attribute-Based Access Control (ABAC'18), Tempe, AZ, USA, 19–21 March 2018; pp. 13–24. [CrossRef]
39. Gupta, M.; Benson, J.; Patwa, F.; Sandhu, R. Secure V2V and V2I Communication in Intelligent Transportation using Cloudlets. *IEEE Trans. Serv. Comput.* **2020**. [CrossRef]
40. Gupta, M.; Awaysheh, F.M.; Benson, J.; Alazab, M.; Patwa, F.; Sandhu, R. An Attribute-Based Access Control for Cloud Enabled Industrial Smart Vehicles. *IEEE Trans. Ind. Inform.* **2021**, *17*, 4288–4297. [CrossRef]
41. Gupta, M.; Sandhu, R. The GURA$_G$ Administrative Model for User and Group Attribute Assignment. In *Network and System Security*; Lecture Notes in Computer Science; Chen, J., Piuri, V., Su, C., Yung, M., Eds.; Springer: Cham, Switzerland, 2016; Volume 9955._21. [CrossRef]
42. Lynn, B. On the implementation of Pairing-Based Cryptosystems. Ph.D. Thesis, Standford University, Stanford, CA, USA, 2007.
43. Wang, J. Java Realization for Ciphertext-Policy Attribute-Based Encryption. 2012. Available online: https://junwei.co/cpabe/ (accessed on 26 December 2020).
44. Ciphertext-Policy Attribute Based Encryption Toolkit. 2018. Available online: http://hms.isi.jhu.edu/acsc/cpabe/ (accessed on 26 December 2020).
45. The Pairing-Based Cryptography Library. 2012. Available online: https://crypto.stanford.edu/pbc/ (accessed on 26 October 2020).
46. Available online: https://health.data.ny.gov/api/views/tsg2-5hds/files/5ded175f-ecf3-4dd2-bb38-df464b137958?filename=NYSDOH_HospitalInpatientDischarges_SPARCS_De-Identified_2016.zip (accessed on 26 October 2020).
47. Wu, A.; Zheng, D.; Zhang, Y.; Yang, M. Hidden Policy Attribute-Based Data Sharing with Direct Revocation and Keyword Search in Cloud Computing. *Sensors* **2018**, *18*, 2158. [CrossRef] [PubMed]
48. Odelu, V.; Das, A.K.; Khan, M.K.; Choo, K.R.; Jo, M. Expressive CP-ABE Scheme for Mobile Devices in IoT Satisfying Constant-Size Keys and Ciphertexts. *IEEE Access* **2017**, *5*, 3273–3283. [CrossRef]
49. Dmitrienko, M.; Hadzic, A.; Lohr, M.; Sadeghi, T.; Winandy, M. On the feasibility of attribute-based encryption on internet of things devices. *IEEE Access* **2016**, *36*, 25–35.
50. Zhang, F.; Safavi-Naini, R.; Susilo, W. An Efficient Signature Scheme from Bilinear Pairings and Its Applications. In Proceedings of the International Workshop on Public Key Cryptography, Singapore, 1–4 March 2004; Volume 2947, pp. 277–290.