Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 1384405, 17 pages
https://doi.org/10.1155/2022/1384405

Research Article

An SKP-ABE Scheme for Secure and Efficient Data Sharing in Cloud Environments

Yong-Woon Hwang,[1] **Su-Hyun Kim**,[2] **Daehee Seo**,[3] **and Im-Yeong Lee**[1]

[1]*Department of Software Convergence, Soonchunhyang University, Asan 31538, Republic of Korea*
[2]*ICT Industry Strategy Team, National IT Industry Promotion Agency, Jincheon-Gun 27872, Republic of Korea*
[3]*Faculty of Artificial Intelligence and Data Engineering, Sangmyung University, Seoul 03016, Republic of Korea*

Correspondence should be addressed to Im-Yeong Lee; imylee@sch.ac.kr

Received 8 April 2022; Revised 15 May 2022; Accepted 20 May 2022; Published 17 June 2022

Academic Editor: Yan Huang

Security threats such as data forgery and leakage may occur when sharing data in cloud environments. Therefore, it is important to encrypt your data and securely access it when sharing it with other users via a cloud server. Of the various security technologies, research on secure data sharing commonly employs Key Policy Attribute-Based Encryption (KP-ABE). However, existing KP-ABE schemes generally lack ciphertext search features. Furthermore, even if a KP-ABE scheme incorporates it, the number of searches required increases markedly by the number of attributes used in the search. It in turn proportionally increases the ciphertext size. In addition, the attribute authority (AA) could be attacked, which can result in the leakage of users' decryption keys. AA is a server that manages user attributes and decryption keys when using attribute-based encryption in a cloud environment. If the AA is curious, it can cause problems with the key escrow with the attributes and decryption (secret) key information of the users it knows. In this paper, to solve all these problems, we present a new scheme called Searchable Key-Policy Attribute-Based Encryption (SKP-ABE) for secure and efficient data sharing in the cloud. This proposed SKP-ABE scheme allows fast ciphertext search and keeps the ciphertext of constant size. The key escrow problem is solved via user key generation.

1. Introduction

Developments in cloud computing technology have made it possible to collect, manage, and share big data from the Internet of Things (IoT)-Cloud environments such as Unmanned Traffic Management (UTM), companies, and the Internet of Medical Things (IoMT). However, as shown in Figure 1, several security threats exist in the cloud [1, 2]. First, cloud service providers cannot be completely trusted. Users think that their data is securely protected if an external cloud is used. However, the service provider may know the data contents stored and utilized on their server. An attacker (a malicious user) can compromise shared data for another security threat. An attacker may access the server, tamper with the stored data, and leak the data. If the data stored on the cloud server is sensitive information, this will pose a significant security threat [3, 4]. Therefore, a security technique that encrypts data stored and transferred in the cloud is required, as is access control for this encrypted data. Of the various security technologies, attribute-based encryption (ABE) ensures secure data encryption/decryption and access control. ABE performs encryption/decryption employing multiple user attributes. It is widely used for secure data sharing in the cloud. ABE schemes include key-policy attribute-based encryption (KP-ABE) and ciphertext-policy attribute-based encryption (CP-ABE). The two ABE schemes depend on the data Access Structure (AS) contained in the ciphertext and the data user secret key. If the AS is included in the ciphertext, the CP-ABE scheme is used, and if the AS is included in the data user secret key, the KP-ABE scheme is

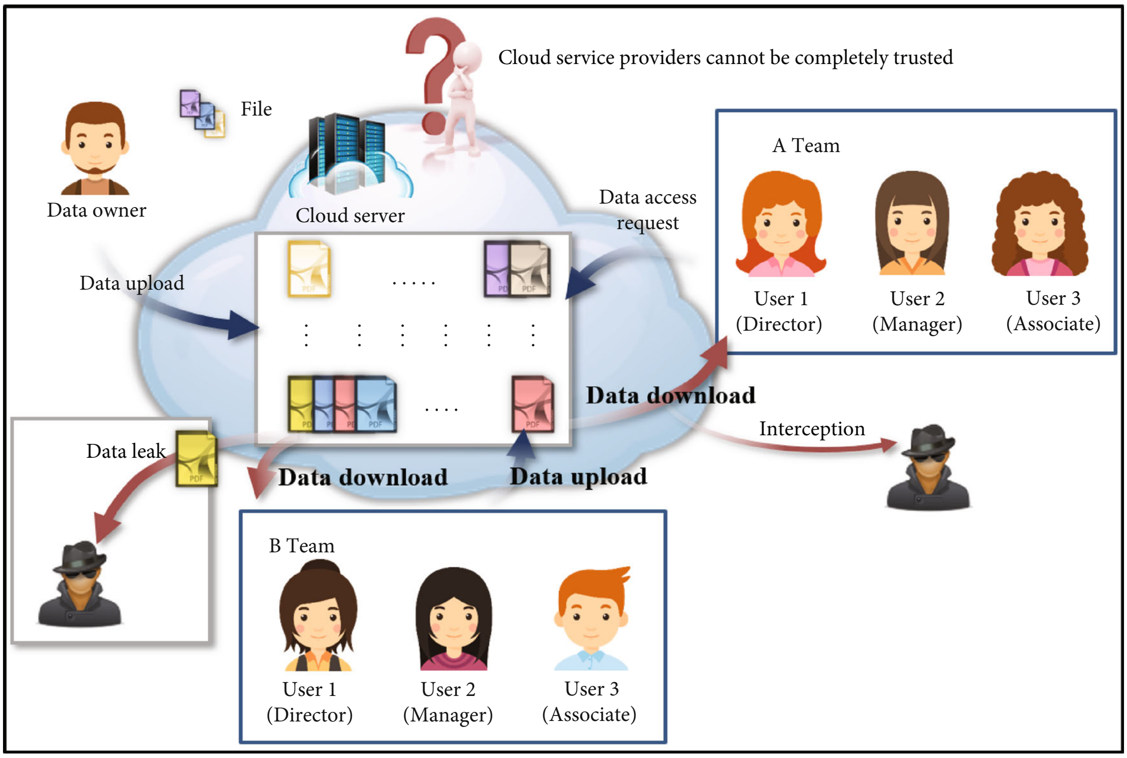


Figure 1: Security threats in the cloud environment.

used. The differences between the two types of ABE schemes are explained in Section 2 [5, 6].

In this paper, we intend to research data sharing in an N:1 cloud environment where data users can decrypt ciphertexts with the attributes of the AS included in the secret key. Here, "N" means multiple users. Since the KP-ABE scheme is suitable in an N:1 cloud environment, research on KP-ABE was conducted. To date, various KP-ABE schemes have been analyzed for secure data storage and sharing technology. However, there are security threats and inefficient schemes among the existing KP-ABE schemes.

First, the traditional KP-ABE schemes encrypt and store data in the cloud that cannot be searched. Therefore, all stored ciphertexts must be decrypted when seeking a desired ciphertext among numerous ciphertexts. This makes the process inefficient. To solve this problem, efforts have been made to introduce searchable encryption [7–10]. However, the number of searches required and the ciphertext size increase proportionally to the number of attributes [11, 12]. This wastes storage space on the server. In addition, when using attribute-based encryption, a server known as the attribute authority (AA) manages user attributes. The AA plays a role in creating secret keys (ciphertext decryption keys) that include public parameters and user attributes. Data owners and users apply the keys to encrypt/decrypt data. If an AA is attacked, users' secret keys may be leaked. Furthermore, most KP-ABE schemes trust their AAs. But still curious AAs can access and decrypt the ciphertexts stored in the cloud with the stored user's attribute information and secret key information. In other words, a key escrow problem may occur by AA [13–15].

In this paper, we propose secure and efficient data storage and sharing system after researching and analyzing ABE to solve the security threats in cloud environments. Our system allows fast ciphertext search, and the ciphertext size is kept constant. The key escrow problem is solved via user key generation. In summary, we establish secure and efficient data storage and sharing system by proposing a searchable key-policy attribute-based encryption (SKP-ABE) system to which various requirements are applied. The contributions of this paper are as follows:

(i) Efficiency of ciphertext search: The cloud server uses searchable encryption technology to quickly search for the ciphertext requested by the user [16, 17]. Compared with existing KP-ABE schemes, this proposed SKP-ABE scheme aggregates the attribute values included in the ciphertext index. In this case, when searching for a ciphertext, it is possible to find the ciphertext in one search regardless of the number of attributes

(ii) Output of ciphertext of constant size: A ciphertext of a constant size is output by aggregating the values of the attributes included in the ciphertext and expressing them as a single value. The size of the ciphertext does not increase according to the number of attributes included in the ciphertext

(iii) Solution of key escrow problem: In existing KP-ABE schemes, the AA generates a key corresponding to the user's AS and transmits it to the user. That is, the AA knows information about the users' secret keys and attributes. It can sufficiently cause a key escrow problem. In this proposed scheme, the AA creates a partial secret key and sends it to the user. The user creates a final secret key with the received partial secret key that can decrypt the ciphertext. Therefore, the AA does not know the users' secret key information, and the key escrow problem that occurs in an AA is solved

The remainder of this paper is organized as follows: Section 2 describes the research background; ABE is explained. It also describes existing KP-ABE schemes and the KP-ABE security model. Section 3 describes the security requirements to be provided. Section 4 describes the proposed SKP-ABE scheme. Section 5 analyzes the security and efficiency of the scheme, and Section 6 concludes the paper.

2. Background

This section describes ABE and the preliminaries and formulas for understanding it. Then, the KP-ABE system and KP-ABE security model are explained.

2.1. Preliminaries

2.1.1. Bilinear Map. Bilinear mapping has been proposed as a tool to attack elliptic curve cryptosystems in the past. However, recently, it has been used as a cryptography tool for information protection, and the algorithms elliptic curve cryptography (ECC), which are based on bilinear mapping, are widely used in IoT environments. A bilinear pairing function is called a bilinear mapping, and the notation is expressed as follows: Suppose we have multiplicative groups $G_1$and G_2 with the same order p. Assume that it is difficult to solve the discrete logarithm problem within a group. Let g be a generator group of G_1, and let $e : G_1 X G_1 \longrightarrow G_2$ be a bilinear mapping that satisfies the following properties:

(1) Bilinearity: For all $P, Q \in G_1$ and all $, b \in Z_p$, $e(P^a, Q^b) = e(P, Q)^{ab}$

(2) Nondegeneracy: For all $Q \in G_1$, if $e(P, Q) = 1$, then $P = 0$

(3) Computability: For all $P, Q \in G_1$, there is an efficient algorithm to compute $e(P, Q) \in G_2$

2.1.2. Bilinear Diffie Hellman (BDH) Assumption. The deterministic BDH assumption means that, given two pairs $(g^a, g^b, g^c, W = e(g, g)^z)$ and $(g^a, g^b, g^c, T = e(g, g)^{abc})$, there is no algorithm A that can distinguish between the two pairs with meaningful probability. Here, $a, b, c, z \in Z_p$. If algorithm A is able to solve the deterministic BDH assumption, that is $|\Pr[A(g^a, g^b, g^c, T) = 1] - \Pr[A(g^a, g^b, g^c, W) = 1]| \geq \epsilon$ if satisfied, then algorithm A has an advantage of ϵ [18].

2.1.3. Bilinear Diffie Hellman Exponent (BDHE) Assumption. The deterministic BDHE assumption means that, given $(h, g, g^{\alpha} \cdots ., g^{\alpha\beta}, g^{\alpha\beta+2}, \cdots g^{\alpha 2\beta})$, there is no algorithm A that can compute $T = e(h, g)^{\alpha\beta+1}$ with a meaningful probability. Here, $h, g \in G_1$, $g_i = g^{\alpha^i} (i = 1, \cdots, 2\beta)$ and $g_{\alpha,\beta} = (g_1, \cdots, g_B, g_{B+2}, \cdots, g_{2B})$; when the next two pairs are $(h, g, g_{\alpha,\beta}, W = e(h, g)^z)$, $(h, g, g_{\alpha,\beta}, T = e(h, g)^{\alpha\beta+1})$. If algorithm A is able to solve the deterministic BDHE assumption, that is $|\Pr[A(h, g, g_{\alpha,\beta}, T) = 1] - \Pr[A(h, g, g_{\alpha,\beta}, W) = 1]| \geq \epsilon$ if satisfied, then algorithm A has an advantage of ϵ [18].

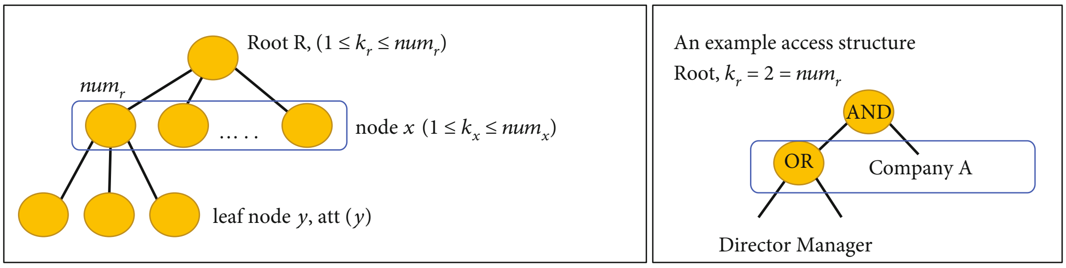
2.1.4. Decisional Bilinear Diffie-Hellman (DBDH) Assumption. Given g^l, g^m, g^n, where $l, m, n \in Z_q$, the DBDH problem is to distinguish g^{lmn} from g^z, where $z \in Z_q$. Given B is an algorithm, and its advantage in solving the problem is $Adv_B^{DBDH}|\Pr[A(g^l, g^m, g^n, g^{lmn}) = 1] - \Pr[A(g^l, g^m, g^n, g^z) = 1]|$. The DBDH assumption states that the advantage of an algorithm B in solving DBDH problem is negligible.

2.1.5. Elliptic Curve Discrete Logarithm Problem (ECDLP) Assumption. Elliptic curve cryptography can achieve the same security as previous public key encryption methods with fewer bits; it is widely used in IoT and other lightweight environments. Compared to the previous public key encryption methods, it uses short keys, so it is easier to manage the keys, and the encryption is processed at high speed. To use ECC, an elliptic curve is a set of solutions (X, Y) of the equation $y^2 = x^3 + ax + b \pmod p$ defined for arbitrary integers a and b. The fact that the point $P = (X, Y)$ is on the elliptic curve means that the previous equation is satisfied. $Q = x \cdot P$ can be defined for any integer x for two points P and Q. Finding the solution x is the discrete logarithm for elliptic curves. That is, it is easy to find Q by using $x \cdot P$ in Q. However, it is very difficult to infer the value of x even if you know Q and P [19].

2.2. Attribute-Based Encryption

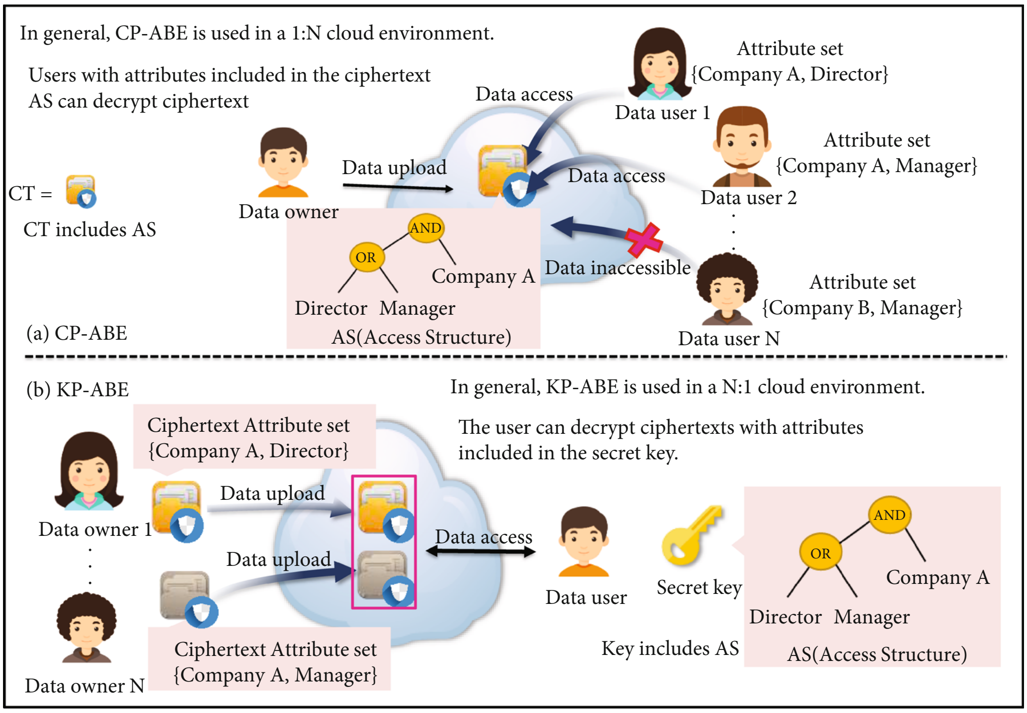
2.2.1. Access Structure. ABE is a scheme of performing encryption/decryption based on an AS created using a set of attributes (e.g., affiliation and occupation) for each entity. Here, the AS is shown in Figure 2. In the access tree, denoted by T, each non-leaf node can represent a threshold gate: an OR gate or an AND gate, depending on the threshold. In general, for all nodes $x \in T$, we use the notations k_x and num_x to represent the threshold of x and the number of children, respectively. For a non-leaf node x, if k_x =1, then x represents an OR gate. If $k_x = num_x$, it represents an AND gate. If $1 < k_x < num_x$, then x is a threshold gate. We define k_x =1 and num_x =0 for leaf node x [5, 6, 20, 21].

2.2.2. Types of ABE. ABE includes CP-ABE or KP-ABE depending on the AS created by the user. In Figure 3(a), the data owner includes the AS when generating the ciphertext and stores it on the cloud server and multiple users can access it. At this time, only if a user's attributes match the attributes of the AS included with the ciphertext can they be decrypted. For example, if the AS is created with {{Director AND Manager} OR Company A}, only users with the Director and Manager attributes among users with the company A attribute can decrypt the ciphertext [5]. The CP-ABE scheme has the advantage of being accessible to any users with the attribute of the AS included in the ciphertext.



Root R, $(1 \leq k_r \leq num_r)$
num_r
node x $(1 \leq k_x \leq num_x)$
leaf node y, att (y)
An example access structure
Root, $k_r = 2 = num_r$
AND
OR
Company A
Director Manager

Figure 2: Description of access structure.



In general, CP-ABE is used in a 1:N cloud environment.
Users with attributes included in the ciphertext AS can decrypt ciphertext
CT =
CT includes AS
Data owner
Data upload
Data access
Data access
Data user 1
Attribute set {Company A, Director}
Data user 2
Attribute set {Company A, Manager}
Data inaccessible
Data user N
Attribute set {Company B, Manager}
AND
OR
Company A
Director Manager
AS(Access Structure)
(a) CP-ABE
(b) KP-ABE
In general, KP-ABE is used in a N:1 cloud environment.
The user can decrypt ciphertexts with attributes included in the secret key.
Ciphertext Attribute set {Company A, Director}
Data owner 1
Data upload
Data upload
Data owner N
Ciphertext Attribute set {Company A, Manager}
Data access
Data user
Secret key
Key includes AS
AND
OR
Company A
Director Manager
AS(Access Structure)

Figure 3: N:N cloud environment using ABE ((a) CP-ABE and (b) KP-ABE).

Therefore, it is widely used in cloud 1:N (N is the number of users) environment. Figure 3(b) shows the KP-ABE scheme. Data users create an AS using their attributes in a KP-ABE scheme and create a corresponding secret, ciphertext decryption key. When data owners generate ciphertexts, they encode the attributes of the users with whom the data will be shared. The ciphertext is stored on the cloud server. Data users can access the cloud server at any later time using a secret key that includes the AS and decrypts the ciphertext with the correct attribute values. For example, if a data owner creates a ciphertext with the attributes {{Director}, {Company A}} and uploads it, only users with the attributes {{Director}, {Company A}} in their AS can decrypt it. In the KP-ABE scheme, when multiple users encrypt data with the attributes of the users who want to share data and upload it to the cloud server, only users with the AS of the attributes designated by the data owner can decrypt the ciphertexts. Therefore, it is widely used in cloud N:1 environment. Figure 3 shows how ABE can be applied to an N: N cloud environment. This paper intends to research a data sharing system in an N:1 cloud environment that allows an authenticated user to decrypt a number of ciphertexts stored with their private key when a large number of data is encrypted and collected and stored. Therefore, research on KP-ABE is suitable.

2.2.3. KP-ABE Model. Figure 4 shows an application of a KP-ABE scheme to cloud environments. There are four entities: an AA, a data owner (users who uploads ciphertext to the cloud), a data user (users who attempts to decrypt ciphertext stored on the cloud), and a cloud storage server. First, a master key and public parameters are generated during the setup phase of the AA. Next, the users create an AS using their attributes, send them to the AA, and request a secret ciphertext decryption key. In a KP-ABE scheme, AS can be created by the user, and an AA can be required to create the AS for the user. In the latter case, the AA generates a secret key corresponding to the user's AS and sends it to the user with the public parameters. When a data owner generates a ciphertext, encryption is performed based on attributes of users that should be allowed access to them. Next, the ciphertext is uploaded and stored on a cloud server. Users registered

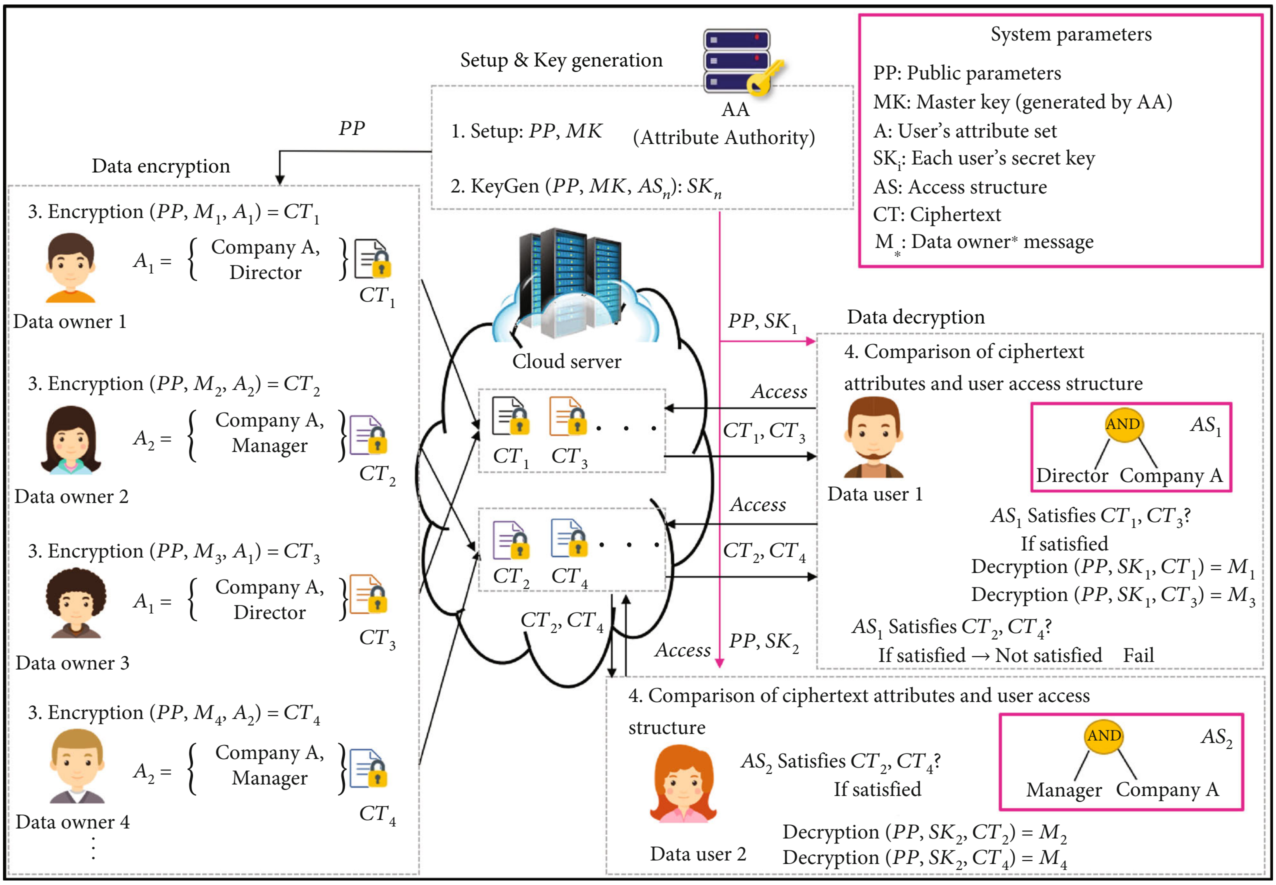

Setup & Key generation
AA
(Attribute Authority)
1. Setup: PP, MK
2. KeyGen (PP, MK, AS_n): SK_n
System parameters
PP: Public parameters
MK: Master key (generated by AA)
A: User's attribute set
SK_i: Each user's secret key
AS: Access structure
CT: Ciphertext
M_*: Data owner* message
PP
Data encryption
3. Encryption (PP, M_1, A_1) = CT_1
A_1 = { Company A, Director }
CT_1
Data owner 1
3. Encryption (PP, M_2, A_2) = CT_2
A_2 = { Company A, Manager }
CT_2
Data owner 2
3. Encryption (PP, M_3, A_1) = CT_3
A_1 = { Company A, Director }
CT_3
Data owner 3
3. Encryption (PP, M_4, A_2) = CT_4
A_2 = { Company A, Manager }
CT_4
Data owner 4
⋮
Cloud server
CT_1 CT_3 . . .
CT_2 CT_4 . . .
CT_2, CT_4
PP, SK_1
Access
CT_1, CT_3
Access
CT_2, CT_4
Access
PP, SK_2
Data decryption
4. Comparison of ciphertext attributes and user access structure
Data user 1
AND
AS_1
Director Company A
AS_1 Satisfies CT_1, CT_3?
If satisfied
Decryption (PP, SK_1, CT_1) = M_1
Decryption (PP, SK_1, CT_3) = M_3
AS_1 Satisfies CT_2, CT_4?
If satisfied → Not satisfied Fail
4. Comparison of ciphertext attributes and user access structure
Data user 2
AS_2 Satisfies CT_2, CT_4?
If satisfied
AND
AS_2
Manager Company A
Decryption (PP, SK_2, CT_2) = M_2
Decryption (PP, SK_2, CT_4) = M_4

Figure 4: Data sharing scheme using KP-ABE in cloud.

in AA generate tokens and send them to the cloud server. The cloud server transmits the ciphertext requested by the users. Finally, the users obtain the data by decrypting the ciphertext using the AS with that attribute and the received secret key [6, 22].

2.3. Challenges to Build KP-ABE Scheme. Various requirements must be provided to build a secure and efficient data sharing system by applying KP-ABE. The requirements are keyword search, constant-size ciphertext output, key escrow problem solving, verifiable outsourcing, attribute withdrawal, AS anonymization, etc. In order to build a secure KP-ABE scheme, research is needed to provide the above-mentioned requirements. However, the KP-ABE scheme is inefficient because the scheme (model) becomes heavy when all requirements are applied. Therefore, there is a need for research to apply the requirements according to the environment.

The SKP-ABE scheme proposed in this paper is also that provides an existing ciphertext search. The difference from the existing KP-ABE scheme, which provides ciphertext search, is to provide a fast ciphertext search by aggregating the attributes included in the token. In addition, it solved the key escrow problem that occurs in AA and provided a ciphertext of a constant size. Therefore, it provides better requirements than the KP-ABE scheme, which provides only the existing keyword search.

2.3.1. Searchable Encryption. As cloud computing develops, users store and manage large amounts of data using storage space provided by an external service provider such as Google cloud. However, when sensitive personal information is stored externally, security issues arise. Therefore, it is important to encrypt all data. However, then the cloud server must decrypt all stored ciphertexts to find data requested by a user. This is very computationally inefficient [7–10]. One of the security technologies to solve this is searchable encryption. Data can be found without decrypting the ciphertext requested by the user. Therefore, when multiple owners encrypt and store data on the cloud, users can efficiently locate the desired ciphertext.

An early version of searchable encryption, proposed by Song, Wagner, and Perrig in 2000, is a hidden search designed to be searchable without leaking plaintext information [23]. However, the initial version lacked a clear definition of security. Since then, searchable encryption systems that use symmetric or asymmetric keys have attracted much attention. Currently, searchable encryption technology is used with ABE to improve ciphertext search efficiency [7–10].

Figure 5 shows a KP-ABE scheme with searchable encryption applied. The existing KP-ABE scheme assumes that when a user requests a ciphertext from the cloud server, the cloud server transmits the ciphertext to the user. However, KP-ABE schemes with searchable encryption add the phase of searching for a ciphertext on the cloud server.

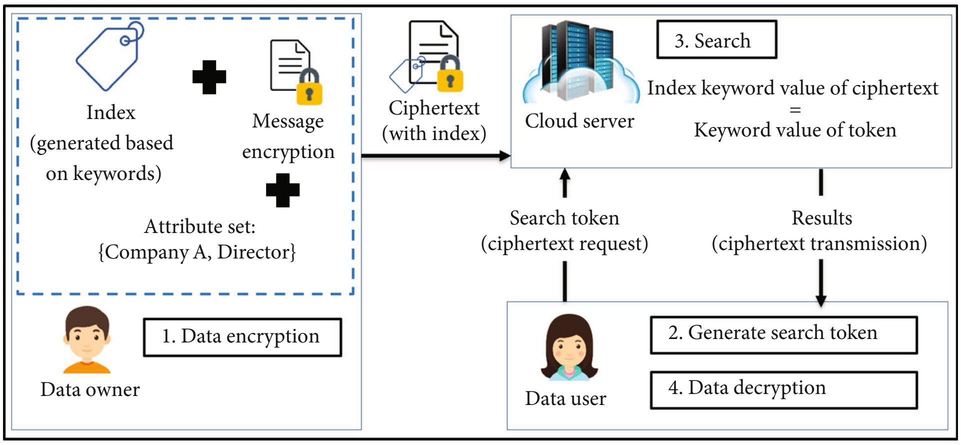


Figure 5: KP-ABE-based data sharing model applying searchable encryption in cloud environment.

In detail, the ciphertext is retrieved from the cloud server based on keywords and attribute values. The data owner selects keywords and attribute values, creates an index, and uploads it to the cloud along with the ciphertext. Next, the user creates a search token using keywords and attribute values to find the ciphertext. Then, it is sent to the cloud server to request the ciphertext. The cloud server searches for the ciphertext by comparing the stored ciphertext index value (including keyword values and attributes) with the search token values (including keyword values and attributes). If matching ciphertexts are found, they are sent to the user. The cloud server finds the requested ciphertext but does not decrypt it [24].

2.3.2. Key Escrow Problem. Key escrow is a system that entrusts encryption (secret) keys to a third party (server) and stores them. If the user key is damaged or lost, the previously entrusted secret key can be issued through the server. However, a server that knows the information about the key may cause a key escrow problem that may attempt to access and decrypt the ciphertext. As a result, user data may be leaked, and various security threats such as abuse of access rights may occur. From the past to the present, in various cryptographic research fields such as key recovery, signature, and ABE, it often occurs in servers (key generation center (KGC) and AA, etc.) that generate and manage keys [25–28]. In an environment where a key escrow problem occurs, it is assumed that users do not completely trust the server managing the key. Therefore, entrusting all key information to the server is a risk factor [29].

The AA is a trusted server that manages properties and generates keys in a data sharing environment using KP-ABE. However, in some KP-ABE schemes, AA is recognized as a semi-trusted server that manages user attributes, so it is mentioned that key escrow problems can occur sufficiently in AA. The term semi-trusted means that the AA is not fully trusted because it has information about the users' secret keys that could cause a key escrow problem. The AAs are honest but curious and have the right to view user information at any time. In the KP-ABE scheme, AA generates a ciphertext decryption key corresponding to the user's attributes and transmits it to the user. Since the AA knows your secret key, it can use it to access the cloud and crack your ciphertext. Therefore, research is being conducted from the existing KP-ABE scheme with single AA to the KP-ABE scheme with multi-AA scheme. This research aims to prevent a key escrow problem in advance with the users' key and attribute information that the AA alone knows [13, 28].

In the multi-AA scheme, when a user requests a secret key by global identity (GID), values corresponding to user attributes are calculated in each AA to create a secret key and send it to the user. Although there is a scheme in which the user generates a secret key with the attribute value received from the AA, usually, multi-AAs generate a secret user key and send it to the user. Above all, since multi-AAs share information about the users' secret key, the AA cannot independently cause a key escrow problem. However, the multi-AA scheme has a disadvantage. The amount of computation required to generate a user secret key increases according to the number of AAs, and a collusion attack between AAs must also be considered. Furthermore, in some KP-ABE schemes, the multi-AA scheme is also viewed as a concept managed by a Central Authority.

2.4. Related Work. In 2006, an initial version of the KP-ABE system was proposed, and based on this, research was conducted to satisfy various requirements. This SKP-ABE scheme provides ciphertext search, constant-size ciphertext, and key escrow problem solving. Table 1 lists an analysis of existing KP-ABE schemes. The description of the KP-ABE scheme that provides the ciphertext search is as follows.

Yin et al. [7] developed a model that adds searchable encryption to the KP-ABE scheme. It is useful when searching for ciphertext in a cloud that manages big data, but the ciphertext size increases with the number of attributes. In addition, as the data owner creates a secret key and transmits it to the user via a secure channel, the data owner knows its secure key. Thus, a key escrow problem may occur.

Ameri et al. [8] considered an environment where the cloud provider was not completely trusted. Their scheme allows the creation of a search token at any time. This token matches all ciphertexts containing the keyword. However, as information leakage is possible, Ameri et al. proposed KP-ABE schemes, in which the search token matches only

Table 1: Comparison of KP-ABE schemes.

KP-ABE scheme	Ciphertext search	Key escrow problem	Ciphertext size
Yin et al. scheme [7]	Provided	Possible to occur	Proportional to the number of attributes
Ameri et al. scheme [8]		Not considered (possible to occur)	
Li et al. scheme [9]			
Meng et al. scheme [10]			
Longo et al. scheme [13]	Not provided	Key escrow problem solved (multi authority)	Constant-size ciphertext
Leyou Zhang et al. scheme [14]		Key escrow problem solved (decentralized authority)	Proportional to the number of attributes
Kai Zhang et al. scheme [30]		Not considered (possible to occur)	Constant-size ciphertext
Belguith et al. scheme [31]			
Goal of the proposal scheme	Provided (fast ciphertext search)	Key escrow problem solved (single authority)	Constant-size ciphertext

ciphertext generated within a specified time interval [8]. That is, it is a scheme that can share ciphertexts within a specified time frame using temporary keywords. Nonetheless, they did not consider the key escrow problem. They assumed that the AA was fully trusted. However, since the AA knows the users' key information, this can cause a key escrow problem. Also, ciphertext size increases by the number of attributes included in the ciphertext.

Li et al. [9] proposed a secured ABE scheme with a searchable encryption function to protect the security and privacy of sensitive data. To counter keyword-guessing attacks, all keywords were signed using secret keys of the data owners when generating ciphertexts. However, depending on the number of attributes, it can increase the size of the ciphertext, and it has the key escrow problem.

Meng et al. proposed a scheme that improved computation efficiency by using a constant-size output ciphertext and a constant pairing operation in a KP-ABE scheme that provides searchable encryption [10]. However, the key escrow problem remained possible.

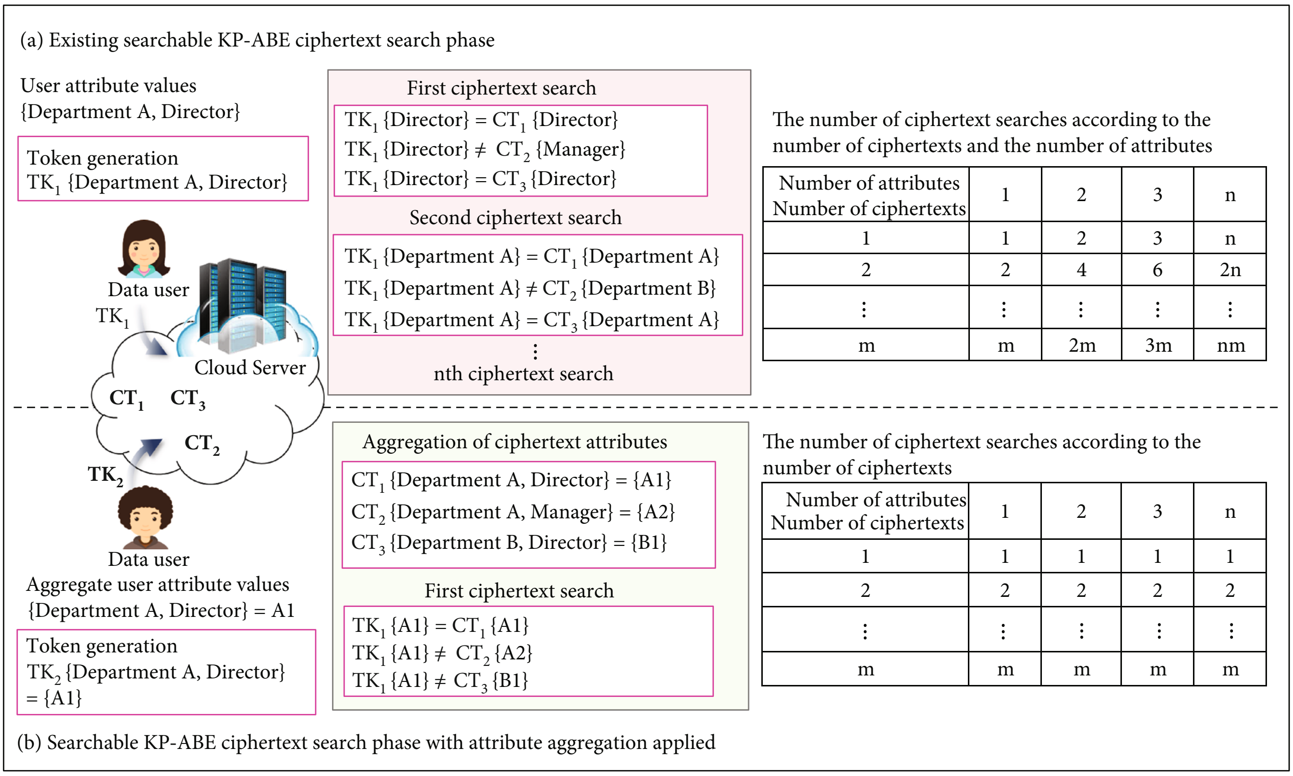
Figure 6 shows how ciphertext is searched on the cloud server. It assumes that three ciphertexts are stored on the cloud server, each with two attributes. When the server searches for a ciphertext, the first search compares the first attribute of the token with the first attribute of the ciphertext. The second search compares the second attribute of the token with the second attribute of the ciphertext and finds a matching ciphertext. In Figure 6(a), the number of ciphertext searches increases proportionally to the number of attributes contained in the token and ciphertexts. For example, the searchable KP-ABE scheme was mentioned above (Yin et al., Ameri et al., Li et al., and Meng et al.). To solve this problem, an aggregate operation is performed on the attribute value included in the ciphertext and the attribute value of the token generated by the user. Then, the aggregated attribute values of the token and the ciphertext are compared to find a matching ciphertext [16, 17, 32]. Figure 6(b) shows the aggregate attributes of tokens and ciphertexts when searching for a ciphertext. As a result, the number of ciphertext searches is not affected by the number of attributes contained in the tokens and ciphertexts. The disadvantage is that tokens can be generated in multiple ways depending on the aggregate attributes of the ciphertext that the user wants to find. However, if an aggregation operation is used, searching for a ciphertext requested by the user on the server will be more efficient than the scheme in Figure 6(a). In terms of decryption, since the goal is to find the ciphertext in most KP-ABE schemes that provide searchable encryption, the decryption process of the ciphertext is omitted. Therefore, partial decryption is not provided.

The KP-ABE scheme that solves the key escrow problem and constant size is as follows. The KP-ABE schemes of Longo et al. [13] and Leyou Zhang et al. [14] solved the key escrow problem using a multi-AA or decentralized AA. By dividing the key generation authority of a single AA among multiple AAs, no individual AA knows all of the information about a users' secret key. However, the ciphertext search function is not provided, and constant-size ciphertext and partial decryption are provided depending on the scheme. The schemes of both Kai Zhang et al. [30] and Belguith et al. [31] output constant-size ciphertext [26, 27].

2.5. KP-ABE Security Model Definition. The security goal of searchable ABE is to prevent an attacker from obtaining information about a keyword from the search token and index keywords in the ciphertext. In other words, if a search token is not found, it should not disclose information about the index keyword w. KP-ABE schemes must provide security against attackers who can obtain search tokens for arbitrary keywords w of their choosing. Even in these attacks, the attacker should not be able to distinguish the encryption of the keyword w_1 and the encryption of the keyword w_0, which does not include obtaining the trapdoor [7, 16]. We use an adaptive chosen keyword attack game and an adaptive chosen plaintext attack game to define the security model of search tokens and index keywords. We provide a formal definition of security through the following games between a probabilistic polynomial-time attacker A and challenger C.

2.5.1. Adaptive Chosen Keyword Attack Game

(1) Challenger C executes Setup(1^λ) to generate master key MK and public parameter PP. Then, it sends the PP to attacker A


(a) Existing searchable KP-ABE ciphertext search phase
User attribute values
{Department A, Director}
Token generation
TK_1 {Department A, Director}
Data user
TK_1
Cloud Server
CT_1 CT_3
CT_2
TK_2
Data user
First ciphertext search
TK_1 {Director} = CT_1 {Director}
TK_1 {Director} ≠ CT_2 {Manager}
TK_1 {Director} = CT_3 {Director}
Second ciphertext search
TK_1 {Department A} = CT_1 {Department A}
TK_1 {Department A} ≠ CT_2 {Department B}
TK_1 {Department A} = CT_3 {Department A}
⋮
nth ciphertext search
The number of ciphertext searches according to the number of ciphertexts and the number of attributes
Aggregate user attribute values
{Department A, Director} = A1
Token generation
TK_2 {Department A, Director} = {A1}
Aggregation of ciphertext attributes
CT_1 {Department A, Director} = {A1}
CT_2 {Department A, Manager} = {A2}
CT_3 {Department B, Director} = {B1}
First ciphertext search
TK_1 {A1} = CT_1 {A1}
TK_1 {A1} ≠ CT_2 {A2}
TK_1 {A1} ≠ CT_3 {B1}
The number of ciphertext searches according to the number of ciphertexts
(b) Searchable KP-ABE ciphertext search phase with attribute aggregation applied

Number of attributes Number of ciphertexts	1	2	3	n
1	1	2	3	n
2	2	4	6	2n
⋮	⋮	⋮	⋮	⋮
m	m	2m	3m	nm

Number of attributes Number of ciphertexts	1	2	3	n
1	1	1	1	1
2	2	2	2	2
⋮	⋮	⋮	⋮	⋮
m	m	m	m	m

FIGURE 6: Existing searchable KP-ABE ciphertext search phase.

(2) Attacker A can adaptively query the ciphertext for all search keywords. Accordingly, when A requests the ciphertext for the search keyword w_b, C generates the ciphertext as $\mathrm{Index}(w_b)$ and sends it to the attacker A

(3) Attacker A selects two keywords w_0 and w_1 and sends them to challenger C. C fairly selects a random bit value as $b \in \{0, 1\}$, has the attribute set $\{s_1 \ldots s_n\}$ received from the attacker, and encrypts it with w_b to generate $\mathrm{Index}(w_b)$. And it sends the ciphertext index to the attacker

(4) Attacker A continuously requests a private key query from challenger C and generates a legitimate search token by encrypting the query keyword w (w is expressed as w_0 or w_1)

(5) Attacker A guesses that b is b'. We define the advantage that attacker A wins in the above game within stochastic polynomial time as $|\Pr[b = b'] - 1/2|$.

Definition 1. Searchable ABE is semantically secure against adaptive chosen keyword attacks in the above security game when the attacker has at most a negligible advantage in probabilistic polynomial time (PPT). That is, in the chosen keyword attack model, the search token and index keyword should not expose the plaintext information of the query keyword.

2.5.2. Adaptive Chosen Plaintext Attack Game

(1) Challenger C executes $\mathrm{Setup}(1^\lambda)$ to generate master key MK and public parameter PP and sends PP to attacker A. A sends a set of attributes $\{s_1, \ldots, s_n\}$ that it wants to test to C

(2) Attacker A requests a secret key query corresponding to the access structure $\{AS_1, \ldots, AS_n\}$from C. At this time, the limitation is that the set of attributes $\{s_1, \ldots, s_n\}$ must not satisfy the access structure $\{AS_1, \ldots, AS_n\}$. Attacker A receives the secret key from C, encrypts the keyword to be queried, and generates a search token

(3) Attacker A selects two messages M_0 and M_1 and sends them to challenger C. C fairly selects a random bit value as $b \in \{0, 1\}$, and encrypts it as $CT(M_b)$ with the attribute set $\{s_1, \ldots, s_n\}$ received from the attacker. And it sends the ciphertext $CT(M_b)$ to the attacker

(4) Attacker A continuously requests the secret key query corresponding to the access structure $\{AS_1, \ldots, AS_n\}$ from challenger C as in (2). Restrictions here are the same as in (2).

(5) Attacker A guesses that b is b'. We define the advantage that attacker A wins in the above game within stochastic polynomial-time as $|\Pr[b = b'] - 1/2|$.

Definition 2. Searchable ABE is semantically secure against adaptive chosen plaintext attacks in the security game above if the attacker has at most a negligible advantage in PPT.

3. Security Requirements

This section describes the requirements in terms of security and efficiency, such as data encryption/decryption and data access for secure and efficient data storage and sharing in the cloud.

(i) Shared data confidentiality and integrity: If data stored and shared in the cloud is in plain text, the data is exposed to various security threats. Therefore, security for the shared data is required, and the confidentiality and integrity of shared data must be ensured. The ciphertext should be decryptable only by legitimate users

(ii) No access for unauthorized users: If anyone can access cloud data, various security threats arise. Thus, access control is required. ABE is a security and access control technology. Only an authenticated user can decrypt accessed data by comparing an attribute value specified by the data owner with the AS attribute value of the user's secure key. Thus, users without the correct attributes cannot decrypt the data even if they access it

(iii) Ciphertext search efficiency: It is difficult for a user to search for the desired data among the numerous ciphertexts stored in the cloud. To search for a ciphertext requested by a user, all stored ciphertexts must be decrypted to check the contents of their data. This is inefficient. Therefore, searchable encryption technology which enables users to search for the requested data without decryption is essential [7–10]. However, in some of the existing schemes, the number of searches increases proportionally to the number of attributes when searching for a ciphertext. Therefore, in the KP-ABE scheme, it is necessary to aggregate the values of the attributes corresponding to the ciphertext keywords. As a result, the user should quickly search for the desired ciphertext

(iv) Constant-size ciphertext: In existing KP-ABE schemes, the size of the generated ciphertext is proportional to the number of included attributes. Cloud storage space is used inefficiently due to the increased ciphertext size [11, 12]. Therefore, it is needed to research in which the size of the ciphertext can be constant output regardless of the number of attributes

(v) The key escrow problem: Since the AA knows information about the users' secret keys, it is cannot be fully trusted because that can cause a key escrow problem. Therefore, it is necessary to reduce AAs secret key generation authority. Specifically, the key escrow problem can be solved by generating a secret key using multiple AAs. For example, a user receives a partial secret key from the AA and generates the final secret key [33, 34].

4. The Proposed SKP-ABE Scheme

In this section, our proposed SKP-ABE scheme is described (see Figure 7). When searching for a ciphertext, the attribute values of the token and ciphertext are aggregated and compared.

Therefore, it is possible to find the requested ciphertext quickly. Furthermore, the key escrow problem on an AA is solved by generating a final ciphertext decryption key using a partial secret key received from the AA. In addition, by using a constant-size ciphertext, the effects of attribute number on ciphertext size are minimized. Finally, the cloud server finds the ciphertext and sends it to the user, and the user decrypts it to obtain data.

4.1. System Model

4.1.1. System Entities

(i) Data Owner: The data owner encrypts data and uploads it to the cloud. The owner generates a ciphertext with the attributes of the users who can access the data. Then, an index is created by selecting keywords that can represent the ciphertext (CT). Finally, the CT and index are uploaded together to the cloud server

(ii) Cloud Server: In general, a cloud server includes a storage server in which data is stored and a server that performs operations. For example, the cloud server stores and manages data. When a user requests ciphertext, the server performs a ciphertexts search using the ciphertext index and token value received from the user. After that, the retrieved ciphertexts are sent to the user

(iii) Attribute Authority: The AAs are honest but curious and have the right to view user information at any time. In this proposed SKP-ABE scheme, the secret key generation phase of the AA is modified to the partial secret key generation phase. In addition, when registering a user, a certificate that can be authenticated is generated and then sent to the user. The certificate is used to verify that the user is registered when the user later accesses the cloud server

(iv) Data User: A data user is an entity that downloads and decrypts ciphertext uploaded to the cloud. The user generates a final secret key (FSK) to decrypt the ciphertext using the PSK received from the AA. In addition, by selecting the keywords of the ciphertext to be found, a token is generated. A user can request ciphertexts from the cloud server with a token. When the user receives the ciphertext from the cloud server, it uses FSK to perform decryption to obtain the ciphertext to obtain the data

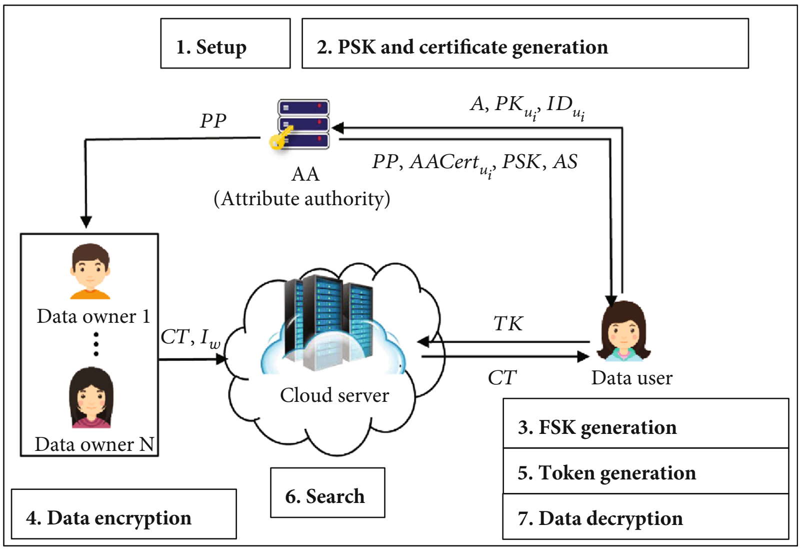


Figure 7: This proposed SKP-ABE scheme scenario

Table 2: Notations.

Symbol	Definition
p, q	Prime order
PP, MK, PK_{AA}	Public parameters, master key, AA's public key
PK_{u_i}, SK_{u_i}	Data user public/private key pair
PSK	Data user partial secret key (partial decryption key)
FSK	Data user final secret key (ciphertext decryption key)
ID_{u_i}	User identifier
$AACert_{u_i}$	Data user certificate
A_U, A	User attribute data, A set of attribute data
AS	Access policy or access structure
$T_{w'}$	Token with keyword w′ (ciphertext search token)
w, w′	Keywords for data owners, keywords for data users
I_w	The ciphertext index value generated based on the keywords
CT	Ciphertext
$H_1(\bullet)$	Cryptographic hash function $\left(\{0,1\}^* \longrightarrow Z_p^*\right)$
$H_2(\bullet)$	Cryptographic hash function $\left(\{0,1\}^* \times \{0,1\}^* \times \{0,1\}^* \longrightarrow Z_p^*\right)$

4.1.2. System Parameters. The system parameters used in the proposed SKP-ABE scheme is shown in Table 2.

4.1.3. Procedure. This proposed SKP-ABE scheme provides secure and efficient data storage and sharing in cloud environments. Compared to existing KP-ABE schemes shown in Table 1, the proposed SKP-ABE scheme meets more requirements. This SKP-ABE scheme consists of 7 phases. The phase are as follows.

(i) Setup(k): The AA generates master key (MK) and public parameters (PP) with security parameter k as input. The data user generates a private/public key pair

(ii) PSK and CertGen(MK, A, PP, ID_{u_i}, PK_{u_i}) ⟶ $AACert_{u_i}$, PSK, AS: When a user requests registration and a partial secret key from the AA, the AA creates AS based on the user's attributes. Next, it generates a partial secret key (PSK) also based on the user's attributes, creates a certificate ($AACert_{u_i}$) based on the user's ID_{u_i} and public key, and sends them all to the user

(iii) FSKGen(PSK, AS) ⟶ FSK: The user receives PSK and AS from the AA and generates the final secret key (FSK) corresponding to the AS

(iv) Encrypt(PP, M, S, w) ⟶ CT, I_w: The data owner selects the message (M) and encrypts with the attribute sets (A) and PP of the users who can access their data. In addition, index value (I_w) is created by selecting keywords that represent the CT and transmitted to the cloud server along with the CT. A keyword is a word that can represent a CT and is known only to the data owner and user

(v) TokGen(FSK, w′) ⟶ TK: The user generates a token $T_{w'}$ to find the CT in the cloud. At this time, a token is generated with the keywords w′ of the CTs to be found and the FSK received from the AA. It then signs the token with the certificate and sends the TK to the cloud to request the CT

(vi) Search(I_w, $T_{w'}$) ⟶ {0, 1}: The cloud server verifies that the CT of the registered user is requested through $AACert_{us_i}$. Then, the CT requested by the user is found using the received $T_{w'}$ and the CT index (I_w). The searched CT is expressed as {0, 1}, and as a result, 0 means not found, and 1 means found. The retrieved ciphertexts are sent to the user

(vii) $\mathrm{Decrypt}(\mathrm{CT}, \mathrm{AS}, \mathrm{l}, \mathrm{FSK}, \mathrm{PP}) \longrightarrow \mathrm{M}$: When the user receives the CTs from the cloud server, it decrypts by comparing the attribute value in the AS with the attribute value in the CT. If the decryption is successful, the user can obtain a message M

4.2. Description of the Proposed SKP-ABE Scheme. The AA generates two cycle multiplication groups G and G_T of prime order p and generates a bilinear map $e : G \times G \longrightarrow G_T$ ($e : G \times G \longrightarrow G_T, \forall i, j \in G, e(i, j) = v, v \in G_T$). Let g denote a generator of G. The AA generates a subgroup G_2 of elliptic curve points of prime order q and chooses a generator P of G_2. Elliptic curve point-based crypto-operations are used to generate user keys, and key security assumes the intractability of ECDLP. Here, the user key means the initially generated key pair (PK_{u_i}, SK_{u_i}) for users to register with AA. Assume that there are n attributes in the universe where the universal set is $A = \{Att_1, Att_2, Att_3, \cdots, Att_n\}$. $W = \{W_1, W_2, W_3, \cdots, W_n\}$ is an AS and includes attributes, such as $Att_i \subset W_i$.

4.2.1. Setup Phase. Initially, the AA creates PP, MK, and PK_{AA} in the setup phase. The AA generates random values $\alpha, k \in Z_p^*$, $t_i \in G$ and computes $f = g^k$, $EP = e(g, g)^{\alpha}$.

The public parameters, master key, and public key are generated as follows:

$$\begin{aligned} PP = {} & <G, G_T, G_2\ e, g, \{T_i = g^{t_i}\}_{i\in[1,n]}, f \\ & = g^k, EP = e(g, g)^{\alpha}, H_1, H_2 >, \end{aligned} \tag{1}$$

$$MK = <\alpha, \{t_i\}_{i\in[1,n]} >, PK_{AA} = <\alpha \bullet P > . \tag{2}$$

The data user selects a random value for $x_{u_i} \in Z_p^*$ and generates a private key/public key pair as follows:

$$PK_{u_i}, SK_{u_i} = <x_{u_i} \bullet P, x_{u_i} > . \tag{3}$$

The user requests registration and a partial secret key by transmitting their attribute set $A = \{Att_1, Att_2, Att_3, \cdots, Att_n\}$, public key (PK_{u_i}), and identifier (ID_{u_i}) to the AA.

4.2.2. PSK and Certificate Generation Phase. The AA creates an access tree AS with a leaf node l value set based on user attributes. And PSK is created with the attribute value corresponding to AS. In addition, the user's public key and ID are used to generate a certificate. The AA sends the PP to the data owner and PP, $AACert_{u_i}$, PSK, and AS to the user:

$$D_{i,j} = g^{t_n A} \text{or}\ g^{t_{n+1} A}, H_{Att_i} = H_1(Att_i)^k{}_{i\in[1,n]}, \tag{4}$$

$$PSK = <g^{\alpha}, \{D_{i,1}\}_{i\in[1,n]}, H_{Att_i} > . \tag{5}$$

When creating a certificate, select $o_{u_i} \in Z_p^*$, $O_{u_i} = o_{u_i} \bullet P$.

$$d_{u_i} = o_{u_i} + \alpha H_2(ID_{u_i}, PK_{u_i}), AACert_{u_i} = (O_{u_i}, d_{u_i}). \tag{6}$$

4.2.3. FSK Generation Phase. Then, the user selects a random value and generates an FSK with the PSK and AS received from AA:

Random number $r_i \in Z_p^*$, $r = \sum_{i=1}^{n} r_i$.

$$D_i = g^{\alpha+r}, D_i' = g^{r_i}, \tag{7}$$

$$FSK = <AS, D_i, D_i', \{D_{i,1}\}_{i\in[1,n]}, H_{Att_i} > \tag{8}$$

4.2.4. Data Encryption Phase. The data owner creates a ciphertext with the PP and the attribute of the user that can access the data. Then, keywords representing the ciphertext are selected, and an index value I_w is generated for the keyword and transmitted together with the CT (see Equations (9)-(12)).

Select message M and add random numbers $s_i, s' \in Z_p^*$, such that $s = \sum_{i=1}^{n} s_i$.

Select attribute set $A = \{Att_1, Att_2, Att_3, \cdots, Att_n\}$ and keyword w (the keyword is a value that indicates the ciphertext created by the data owner and requested by the user. A ciphertext index can use a single keyword, and multiple keywords are more secure).

$$C_0 = M \bullet EP^s, C_1 = h^s, \tag{9}$$

$$C_2 = g^s \bullet \prod_{i\in n} g^{t_i Att_i}, C_3 = \prod_{i=1}^{n} H_1(Att_i)^s, \tag{10}$$

$$\tilde{C}_1 = e\left(f, g^{ws'}\right), \tilde{C}_2 = g^{ss'}, \tag{11}$$

$$CT = <A, C_0, C_1, C_2, C_3 >, I_w = <\tilde{C}_1, \tilde{C}_2, C_3 > . \tag{12}$$

An index value of I_w is set for each CT. The data owner sends CT and I_w to the cloud server. The cloud server securely stores the CT and I_w received from the data owner.

4.2.5. Token Generation Phase. The user selects a keyword w' in the ciphertext to found. Then, the user generates a token $T_{w'}$ using FSK that can be used to find a ciphertext. After token generation, the token is signed with the certificate received from AA (see Equations (13) and (14)).

Select a keyword to search for and generate a token.

$$T_{w'} = e\left(\prod_{i=1}^{n} H_{Att_i}, g^{w'}\right). \tag{13}$$

Sign using a certificate:

$$AACert_{us_i} = d_{u_i} + x_{u_i} H_2(ID_{u_i}, PK_{u_i}, T_{w'}). \tag{14}$$

The user requests a ciphertext by sending $TK = (T_{w'}, AACert_{us_i})$ to the cloud server.

4.2.6. Search Phase. The cloud server verifies the registered user *i* and token through $AACert_{us_i}$. After verification, the $T_{w'}$ received from the user is compared to the I_w of the CTs stored on the server, and matching ciphertexts are found. This will only happen when the keyword w' selected by the user and the keyword w selected by the data owner are the same. The search result is displayed as $\{0, 1\}$. The retrieved ciphertexts are sent to the user:

$$\begin{aligned} \mathrm{AACert}_{\mathrm{us}_i} \bullet \mathrm{P} &= \mathrm{o}_{\mathrm{u}_i} \bullet \mathrm{P} + \alpha \bullet \mathrm{P} * \mathrm{H}_2\left(\mathrm{ID}_{\mathrm{u}_i}, \mathrm{PK}_{\mathrm{u}_i}\right) \\ &\quad + \mathrm{x}_{\mathrm{u}_i} \bullet \mathrm{P} * \mathrm{H}_2\left(\mathrm{ID}_{\mathrm{u}_i}, \mathrm{PK}_{\mathrm{u}_i}, \mathrm{T}_{\mathrm{w}'}\right) \\ &= \mathrm{O}_{\mathrm{u}_i} + \mathrm{PK}_{\mathrm{AA}} * \mathrm{H}_2\left(\mathrm{ID}_{\mathrm{u}_i}, \mathrm{PK}_{\mathrm{u}_i}\right) \\ &\quad + \mathrm{PK}_{\mathrm{u}_i} * \mathrm{H}_2\left(\mathrm{ID}_{\mathrm{u}_i}, \mathrm{PK}_{\mathrm{u}_i}, \mathrm{T}_{\mathrm{w}'}\right), \end{aligned} \tag{15}$$

Ciphertext search: $\mathrm{e}(\tilde{\mathrm{C}}_2, \mathrm{T}_{\mathrm{w}'}) = \mathrm{e}(\tilde{\mathrm{C}}_1, \mathrm{C}_3)$

$$\begin{aligned} \mathrm{e}\left(\tilde{\mathrm{C}}_2, \mathrm{T}_{\mathrm{w}'}\right) &= \mathrm{e}\left(\mathrm{g}^{\mathrm{ss}'}, \mathrm{e}\left(\prod_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{H}_{\mathrm{Att}_i}, \mathrm{g}^{\mathrm{w}'}\right)\right) \\ &= \mathrm{e}\left(\mathrm{e}\left(\mathrm{g}^{\mathrm{s}'\mathrm{w}}, \mathrm{g}^{\mathrm{k}}\right), \prod_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{H}_1(\mathrm{Att}_{\mathrm{i}})^{\mathrm{s}}\right) = \mathrm{e}\left(\tilde{\mathrm{C}}_1, \mathrm{C}_3\right). \end{aligned} \tag{16}$$

4.2.7. Data Decryption Phase. The user performs decryption by comparing the attribute value specified in the user AS with the attribute values included in the ciphertext. Parameter l refers to the attribute value (leaf-node) of the user AS. If the decryption is successful, the users obtain M (see Equations (17) and (18)).

Access structure $\mathrm{W} = \{\mathrm{W}_1, \mathrm{W}_2, \mathrm{W}_3, \cdots, \mathrm{W}_\mathrm{n}\}$.

If $\mathrm{Att}_\mathrm{i} \in \mathrm{W}_\mathrm{i}$, compute $\mathrm{D}_{\mathrm{i,j}} = (\mathrm{g}^{\mathrm{t}_\mathrm{n}})^{\mathrm{l}_\mathrm{n}}$.

If $\mathrm{Att}_\mathrm{i} \notin \mathrm{W}_\mathrm{i}$, compute $\mathrm{D}_{\mathrm{i,j}} = (\mathrm{g}^{\mathrm{t}_{\mathrm{n}+1}})^{\mathrm{l}_\mathrm{n}}$.

$$\begin{aligned} \mathrm{C} &= \frac{\mathrm{e}\left(\mathrm{C}_1, \prod_{\mathrm{j}\in \mathrm{A}} \mathrm{D}_{\mathrm{i,j}}\right)}{\mathrm{e}\left(\mathrm{C}_2, \mathrm{f}\bullet\prod_{\mathrm{j}\in \mathrm{A}} \mathrm{D}_\mathrm{i}{}^{'}\right)} = \frac{\mathrm{e}\left(\mathrm{C}_1, \prod_{\mathrm{j}\in \mathrm{A}} (\mathrm{g}^{\mathrm{t}_\mathrm{n}})^{\mathrm{l}_\mathrm{n}}\right)}{\mathrm{e}\left(\mathrm{C}_2, \left(\mathrm{g}^\mathrm{k}\bullet\prod_{\mathrm{j}\in \mathrm{A}} \mathrm{g}^{\mathrm{r}_\mathrm{i}}\right)\right)} \\ &= \frac{\mathrm{e}\left(\mathrm{g}^{\mathrm{ks}}, \prod_{\mathrm{j}\in \mathrm{A}} (\mathrm{g}^{\mathrm{t}_\mathrm{n}})^{\mathrm{l}_\mathrm{n}}\right)}{\mathrm{e}\left(\left(\mathrm{g}^\mathrm{s}\bullet\prod_{\mathrm{j}\in \mathrm{A}} (\mathrm{g}^{\mathrm{t}_\mathrm{n}})^{\mathrm{l}_\mathrm{n}}\right), \left(\mathrm{g}^\mathrm{k}\bullet\prod_{\mathrm{j}\in \mathrm{A}} \mathrm{g}^{\mathrm{r}_\mathrm{i}}\right)\right)}, \\ &\frac{\mathrm{e}\left(\mathrm{g}^{\mathrm{ks}}, \mathrm{g}^{\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{t}_\mathrm{n} \mathrm{l}_\mathrm{n}}\right)}{\mathrm{e}\left(\left(\mathrm{g}^\mathrm{s}, \mathrm{g}^{\sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{t}_\mathrm{n} \mathrm{l}_\mathrm{n}}\right), \left(\mathrm{g}^\mathrm{k}, \mathrm{g}^\mathrm{r}\right)\right)} = \frac{\mathrm{e}(\mathrm{g}, \mathrm{g})^{\mathrm{ks}}}{\mathrm{e}(\mathrm{g}, \mathrm{g})^{\mathrm{ks}+\mathrm{sr}}} = \mathrm{e}(\mathrm{g}, \mathrm{g})^{-\mathrm{sr}}, \end{aligned} \tag{17}$$

$$\begin{aligned} \mathrm{M} &= \frac{\mathrm{C}_0}{\mathrm{e}(\mathrm{C}_1, \mathrm{D}_\mathrm{i})\bullet\mathrm{C}} = \frac{\mathrm{M}\bullet\mathrm{EP}^\mathrm{s}}{\mathrm{e}(\mathrm{g}^\mathrm{s}, \mathrm{g}^{\alpha+\mathrm{r}})\bullet\mathrm{e}(\mathrm{g}, \mathrm{g})^{-\mathrm{sr}}} \\ &= \frac{\mathrm{M}\bullet\mathrm{e}(\mathrm{g}, \mathrm{g})^{\alpha s}}{\mathrm{e}(\mathrm{g}, \mathrm{g})^{\alpha s+rs-rs}}. \end{aligned} \tag{18}$$

5. Analysis of Proposed SKP-ABE Scheme

This proposed SKP-ABE scheme was analyzed for security and efficiency to satisfy the security requirements detailed in Section 3. Table 3 is an analysis table comparing the existing scheme and the proposed SKP-ABE scheme in terms of security and efficiency.

5.1. Security Analysis

(i) Shared data confidentiality and integrity: Data confidentiality and integrity are protected because data are encrypted, stored, and shared using a KP-ABE scheme. Data is encrypted using attributes $A = \{Att_1, Att_2, Att_3, \cdots, Att_n\}$. Therefore, only a user with an AS matches the attributes for the ciphertext and has the corresponding FSK can decrypt and obtain the data. An attacker who steals data cannot decrypt it

(ii) Access control: In the existing KP-ABE scheme, if the user had the secret key received from AA, the user could create a token and request a ciphertext by accessing the cloud. It is possible to access the cloud and request a ciphertext without further authentication. Ciphertext is decrypted using the user attributes. However, if anyone can access the cloud server, it is difficult to restrict the users. If anyone can access the cloud server, it is difficult to restrict the users. Furthermore, data theft or forgery may occur if a user is malicious. Therefore, an access control function that ensures that only registered users can access the cloud server is required. In the proposed SKP-ABE scheme, only users registered by the AA can access the cloud and request a ciphertext. Each registered user receives an $AACert_{u_i}$ from the AA. The cloud server verifies the validity of $AACert_{us_i} \bullet P = O_{u_i} + PK_{AA} * H_2(ID_{u_i}, PK_{u_i}, T_{w'})$ using the user's public key PK_{u_i} and ID_{u_i}, $T_{w'}$ and O_{u_i}; the ciphertext search is performed. Then, the found ciphertext is sent to the user. Therefore, unauthorized users or third parties cannot access the cloud server other than registered users

(iii) Key escrow problem: To solve the key escrow problem, the AA does not know the users' secret key information completely. In our proposed SKP-ABE system, the user receives a partial secret key from AA and generates a final secret key. The value $(D_i = g^{\alpha+r}, D_i{}' = g^{r_i})$ included in the final secret key is a value required for the users to decrypt data, and only the user who generated the final secret key knows. In the SKP-ABE system, when requesting a ciphertext, an access token signed with a certificate is required, so AA cannot generate it and therefore cannot request a ciphertext. If it is assumed that the AA acquires the user's partial secret key and search token and accesses the cloud, it can search for and attempt to decrypt the ciphertext but cannot finally decrypt the ciphertext. This scheme is similar to solving the key escrow problem from KGC in the certificate-based signature. It was applied to our proposed scheme. In the phase where AA issues $AACert_{u_i}, PSK, AS$ to the user, even if the attacker obtains $AACert_{u_i}, PSK, AS$, the attacker cannot access the cloud with the obtained certificate because he does not know the users' private key. In general, an ABE scheme assumes that the

Table 3: Comparison between the Proposed SKP-ABE scheme and the existing KP-ABE scheme.

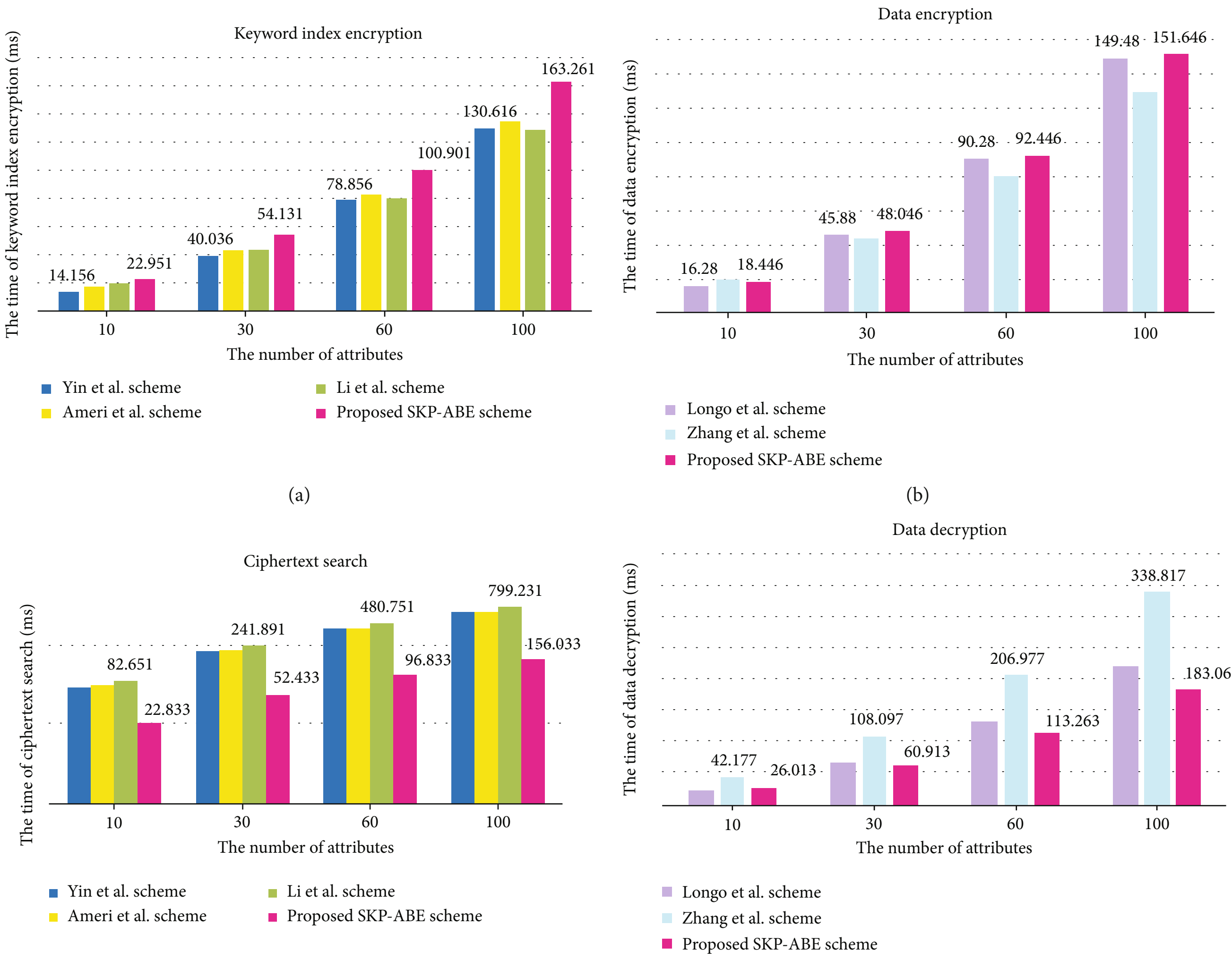
	Yin et al. scheme [7]	Ameri et al. scheme [8]	Li et al. scheme [9]	Longo et al. scheme [13]	Zhang et al. scheme [14]	Proposed scheme
Ciphertext search	Provided			Not provided		Provided
Number of searches	Number of searches increases by the number of attributes					Only one search is required because of attribute aggregation (fast search)
Key escrow problem	Key escrow issues not considered (key escrow problems may occur enough)			Key escrow problem solved (multiauthority)		Key escrow problem solved (user generates decryption key)
Ciphertext size	Proportional to the number of attributes			Constant-size ciphertext	Proportional to the number of attributes	Constant-size ciphertext
Secret key (ciphertext decryption)	$3nT_E$	$3nT_E + nH$	$(4n+1)T_E + 2nT_M$	$K((n+1)T_E + 3nT_M)$	$(2n+2K+5)T_E + 2T_M$	$(2n+2)T_E + nH + (n-1)T_M + E$
Encryption (ciphertext)	—			$(nK+1)T_M +$ $(n+1)T_E$	$(2K+n+2)T_E +2$ $(K-1)T_M + kP + M$	$(n+3)T_E + nT_M + M$
Encryption (ciphertext index)	$(n+1)T_E$ $+ E + nH$	$(n+4)T_E +$ $(n+2)H + 5M$	$(n+6)T_E + T_M + 2H$	Not provided(this is index encryption search)		$P + (n+4)T_E + (n-1)T_M + nH$
Search (test)	$2nP + nE$	$(2n+1)P + lE$	$(2n+1)P + nT_E + nT_M$			$3P + (n-1)T_M + nT_E$
Decryption (user)	—			$nP + nT_E$+(n-1) T_M	$(K+1+n)P + nE$	$3P + (2n+1)T_M + nT_E$

P: pairing operation; M: multiplication operation; E: exponentiation operation; n: number of attributes; H: hash function; T_E : *Exponentiation* in G; T_M : *Multiplication* in G; K : *Number* of *Attribute Authority*.

communication channel between the AA and the data owner and between the AA and the data user is a secure

(iv) Protection against chosen keyword attacks using a secure game model: The proposed SKP-ABE scheme counters a selectively chosen keyword attack game performed by an attacker if the DBDH assumption is valid. In the secure game model, attacker A can adaptively query the ciphertext for all search keywords. In that case, the plaintext of an index keyword is not exposed. In the security game model, it is assumed that probabilistic polynomial time attacker A and simulator B communicate with each other. Simulator B executes $Setup(1^\lambda)$ generates master key $(MK = <\alpha, \{t_i\}_{i\in[1,n]} >)$, and public parameter $(PP = <G, G_T, G_2\ e, g, \{T_i = g^{t_i}\}_{i\in[1,n]}, f = \quad g^k, EP = e(g,g)^\alpha, H_1, H_2 >)$, and sends PP to attacker A. A sends the attributes set $A = \{Att_1, Att_2, Att_3, \cdots, Att_n\}$; it wants to challenge to B. Attacker A requests the ciphertext index for the search keyword w' from B, and B outputs the ciphertext index (Index(w')). Then, it sends the output value to A. Attacker A selects two keywords w_0 and w_1 and sends them to B. B fairly selects a random bit of b ∈ {0,1}, has the attribute set $A = \{Att_1, Att_2, Att_3, \cdots, Att_n\}$ and w from the attacker, and outputs the corresponding value $I_{(w_b)} = <\tilde{C}_1, \tilde{C}_2, C_3 > \longleftarrow (\tilde{C}_1 = e(f, g^{w_b s'}), \tilde{C}_2 = g^{ss'}, C_3 = \prod_{i=1}^n H_1(Att_i)^s)$. Attacker A continuously requests a partial secret key query from B as in 2) and generates a final secret key. $(PSK = <g^\alpha, \{D_{i,1}\}_{i\in[1,n]}, H_{Att_i}> \longrightarrow FSK = <AS, D_i, D_i', \{D_{i,1}\}_{i\in[1,n]}, H_{Att_i} >)$. Then, by selecting the query keywords $w_{b'}$, a valid search token $T_{w_{b'}} = e(\prod_{i=1}^n H_{Att_i}, g^{w_{b'}})$ is continuously generated. Attacker A extracts b from $I_{(w_b)}$ with $T_{w_{b'}}$. However, it is difficult for an attacker to guess $b = b'$. Thus, the system is secure against selective chosen keyword attacks because the attacker finds it very difficult to win the game within probabilistic polynomial time. That is, it is difficult to guess the keyword plaintext information with the ciphertext index value created by Simulator B

(v) Protection against adaptively chosen plaintext attacks using a secure game model: The proposed SKP-ABE scheme counters an adaptively chosen plaintext attack game performed by an attacker if the DBDH assumption is valid. In the secure game model, attacker A can adaptively query the ciphertext for the selected plaintext and communicate with simulator B with each other. Simulator B executes $Setup(1^\lambda)$ generates MK and PP, the same as the chosen keyword attacks security game model. Attacker A requests a partial secret key query corresponding to the access structure $\{AS_1, \ldots, AS_n\}$ from B. At this time, the limitation is that the attribute set $A = \{Att_1, Att_2, Att_3, \cdots, Att_n\}$ must not satisfy the access structure $\{AS_1, \cdots, AS_n\}$. Attacker A receives the partial secret key $(PSK = <g^\alpha, \{D_{i,1}\}_{i\in[1,n]}, H_{Att_i}$



(a)

(b)

(c)

(d)

Figure 8: Comparison of the calculation amount of the existing KP-ABE scheme and this proposed scheme.

$>$) from B and generates a final secret key ($FSK = <AS, D_i, D_i', \{D_{i,1}\}_{i\in[1,n]}, H_{Att_i} >$). Attacker A selects two messages M_0 and M_1 and sends them to B. B fairly selects a random bit of $b \in \{0, 1\}$ and outputs the corresponding ciphertext $CT(M_b)$ with the attribute set $A = \{Att_1, Att_2, Att_3, \cdots, Att_n\}$ and w. And it sends the ciphertext $CT(M_b)$ to the attacker. Attacker A continuously requests the partial secret key query corresponding to the access structure $\{AS_1, \cdots, AS_n\}$ from B. Attacker A extracts b' from $CT(M_b)$. However, it is difficult for an attacker to guess $b = b'$. In other words, the system is selectively secure against adaptively chosen plaintext attacks, because the attacker finds it very difficult to win the game within the probabilistic polynomial time. It is difficult to guess the plaintext information through the ciphertext created by Simulator B

5.2. Efficiency. The computational amount measurements shown in Figure 8 were performed using a Windows system equipped with a 3.50GHz Intel Core i5-4690 processor and 8GB of RAM. Pairing calculations used the pairing-based cryptographic library available at [35]. ECC implementation used the Koblitz elliptic curve $y^2 = x^3 + ax + b \pmod{p}$ with $a = 1$ and $b = 1$ and the $163 - bit$ random prime defined as $F_{2^{163}}$. The proposed scheme includes a process of aggregating attributes in the encryption phase. Therefore, it can be seen from Figure 8 that the amount of computation required for keyword index encryption (a) and data encryption (b) is larger than that of the existing KP-ABE scheme. However, ciphertext search performance (c) and ciphertext decryption performance (d) are more efficient than the existing KP-ABE scheme. Therefore, the proposed SKP-ABE scheme efficiently provides ciphertext search and the user's ciphertext decryption performance. In order to compare the amount of computations in the same environment, one AA was assumed for the scheme of Longo et al. [13], and the scheme of Zhang et al. [14], when the calculation were performed.

(i) Efficient ciphertext search: When a user requests a ciphertext stored on the cloud server using keywords, search is generally inefficient because the server decrypts all ciphertexts to find required data. Accordingly, we implement searchable encryption, which allows users to search for a requested ciphertext without having to decrypt the ciphertext. However, such schemes still suffer from several problems, as discussed above. Therefore, in our proposed SKP-ABE scheme, to address inefficient searching, the parameters of index I_w, that is, the attribute values corresponding to keywords in $C_3 = \prod_{i=1}^{n} H_1(Att_i)^k$, are aggregated and expressed a single value. The attribute values included in the token $T_{w'} = e(\prod_{i=1}^{n} H_1(Att_i)^k, g^{w'})$ are also aggregated and expressed as one value. Thus, if the attributes are {{Director}, {Company A}}, this can be expressed as {{Director}, {Company A}} = $C_3 = \prod_{i=1}^{n} H_1(Att_i)^k$. The ciphertext search seeks matches to C_3 regardless of the number of attributes. This is faster than the existing analyzed KP-ABE schemes because the number of searches is reduced as the number of attributes is irrelevant. Because the values of attributes are pre-aggregated, the user rapidly finds the required ciphertext

(ii) Constant-size ciphertext: In existing KP-ABE schemes, the size of the ciphertext increases in proportion to the number of attributes specified when generating a ciphertext. For example, in Yin et al.'s scheme, it can be seen through I(w) = $(A, I' = \mathrm{e}(\mathrm{g}_1, \mathrm{g}_\mathrm{i})^{\mathrm{sH(w)}}, \mathrm{I}'' = \mathrm{g}^\mathrm{s}, \forall \alpha \in \mathrm{A} : \mathrm{I}_\mathrm{a} = \mathrm{T(a)}^\mathrm{s})$ that the size of the ciphertext increases according to the number of attributes in $\forall \alpha \in \mathrm{A} : \mathrm{I}_\mathrm{a} = \mathrm{T(a)}^\mathrm{s}$. The size of the ciphertext varies depending on the attribute value I_a of 1 or a. In this proposed scheme, to provide a ciphertext of a constant size, the attribute values Att_i included in the ciphertext are aggregated and expressed as one value of $\mathrm{C}_3 = \prod_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{H}_1(\mathrm{Att}_\mathrm{i})^\mathrm{k}$. Regardless of whether the number of attributes Att_i is 1 or i, all are all expressed as C_3. Therefore, it is possible to solve the problem that the number of existing attributes affects the size of the ciphertext. This only affects the ciphertext size, and since the attribute-based aggregation operation is performed in the data encryption phase, the disadvantage is that the amount of data encryption is large compared to the existing KP-ABE scheme

(iii) Efficiency of ciphertext decryption computations: In Table 3, several of the existing schemes (Yin et al., Ameri et al., and Li et al.) do not perform a decryption operation. Therefore, our proposed SKP-ABE scheme is compared with the scheme of Longo et al. and the scheme of Zhang et al., for decryption performance. As shown in Figure 8(d), the cost of decryption by the users is decreased compared to existing schemes (Longo et al., and Zhang et al.). Also, since the two schemes have the disadvantage that the decryption performance increases according to the number of AA, the efficiency of the proposed SKP-ABE scheme is better in terms of the user's decryption cost.

6. Conclusions

In this paper, we proposed an SKP-ABE system for secure and efficient data sharing in cloud environments. The proposed SKP-ABE scheme guarantees data confidentiality and integrity. Those who lack access rights are blocked. Specifically, the attribute value included in the token and the attribute value of the ciphertext are aggregated, and the ciphertext is searched using the aggregated value. As a result, since the number of ciphertext searches is not affected by the number of attributes, ciphertext searches can be performed quickly. Compared with the existing searchable KP-ABE schemes (Yin et al., Ameri et al., and Li et al.), the computation is efficient in terms of the number of ciphertext searches. In addition, when the data owner generates the ciphertext, the size of the ciphertext can be constant output without being proportional to the number of attributes by aggregating the values of the attributes included in the ciphertext. Finally, to solve the key escrow problem in AA, the user receives the PSK from the AA and generates the FSK in this proposed scheme. As a result, since the AA does not know information about the users' FSK, a key escrow problem cannot occur. Therefore, even if you try to decrypt the ciphertext stored in the cloud with only the users' PSK, data cannot be obtained. Compared to the existing KP-ABE scheme (Longo et al. and Zhang et al.) using multiple AA, the proposed scheme has better decryption performance efficiency.

The proposed SKP-ABE scheme is applied to N:1 cloud environment where a large number of data owners and a small number of data users share data. The scheme can be applied in various IoT-cloud environments, such as data sharing between nurses, doctors, and patients in a medical environment data sharing collected by drones in a UTM environment [36–38]. The shared data is secured because only authenticated users have access.

In the future, for the expansion of the proposed SKP-ABE scheme, additional research that can provide the requirements (security and efficiency) considered by KP-ABE is needed. Additionally, a signature and verification phase is required to decrypt the data user and verify that the owner uploads the data obtained.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there is no conflict of interests regarding the publication of this paper.

Authors' Contributions

Yong-Woon Hwang and Su-Hyun Kim contributed equally to this work.

Acknowledgments

This research was supported by the Republic of Korea's MSIT (Ministry of Science and ICT), under the High-Potential Individuals Global Training Program (2021-0-01516) supervised by the IITP (Institute of Information and Communications Technology Planning & Evaluation), and this work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2022R1A2B5B01002490) and the Soonchunhyang University Research Fund.

References

[1] R. Kumar and R. Goyal, "On cloud security requirements, threats, vulnerabilities and countermeasures: a survey," *Computer Science Review*, vol. 33, pp. 1–48, 2019.

[2] A. Singh and K. Chatterjee, "Cloud security issues and challenges: a survey," *Journal of Network and Computer Applications*, vol. 79, pp. 88–115, 2017.

[3] X. Zheng, Z. Cai, and Y. Li, "Data linkage in smart Internet of Things systems: a consideration from a privacy perspective," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 55–61, 2018.

[4] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 1–590, 2016.

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

[6] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *Proceedings of the 13th ACM conference on Computer and communications security*, pp. 89–98, Alexandria, Virginia, USA, 2006.

[7] H. Yin, Y. Xiong, J. Zhang, L. Ou, S. Liao, and Z. Qin, "A key-policy searchable attribute-based encryption scheme for efficient keyword search and fine-grained access control over encrypted data," *Electronics*, vol. 8, no. 3, p. 265, 2019.

[8] M. H. Ameri, M. Delavar, J. Mohajeri, and M. Salmasizadeh, "A key-policy attribute-based temporary keyword search scheme for secure cloud storage," *IEEE Transactions on Cloud Computing*, vol. 8, no. 3, pp. 660–671, 2020.

[9] J. Li, M. Wang, Y. Lu, Y. Zhang, and H. Wang, "ABKS-SKGA: attribute-based keyword search secure against keyword guessing attack," *Computer Standards & Interfaces*, vol. 74, no. 103471, pp. 103471–103477, 2021.

[10] R. Meng, Y. Zhou, J. Ning, K. Liang, J. Han, and W. Susilo, "An efficient key-policy attribute-based searchable encryption in prime-order groups," in *International Conference on Provable Security*, pp. 39–56, Xi'an, China, 2017.

[11] C. J. Wang and J. F. Luo, "A key-policy attribute-based encryption scheme with constant size ciphertext," in *2012 Eighth International Conference on Computational Intelligence and Security*, pp. 447–451, Guangzhou, China, 2012.

[12] J. Lai, R. H. Deng, Y. Li, and J. Weng, "Fully secure key-policy attribute based encryption with constant-size ciphertexts and fast decryption," in *Proceedings of the 9th ACM symposium on Information, computer and communications security*, pp. 239–248, Kyoto, Japan, 2014.

[13] R. Longo, C. Marcolla, and M. Sala, "Collaborative multi-authority KPABE for shorter keys and parameters," *IACR Cryptology. ePrint Archive*, vol. 262, pp. 1–23, 2016.

[14] L. Zhang, P. Liang, and Y. Mu, "Improving privacy-preserving and security for decentralized key-policy attributed-based encryption," *IEEE Access*, vol. 6, pp. 12736–12745, 2018.

[15] Y. Song, H. Wang, X. Wei, and L. Wu, "Efficient attribute-based encryption with privacy-preserving key generation and its application in industrial cloud," *Security and Communication Networks*, vol. 1155, Article ID 3249726, 2019.

[16] H. Wang, X. Dong, and Z. Cao, "Multi-value-independent cipher text policy attribute-based encryption with fast keyword search," *IEEE Transactions on Services Computing*, vol. 13, no. 6, pp. 1142–1151, 2020.

[17] H. Wang, X. Dong, Z. Cao, and D. Li, "Secure and efficient attribute-based encryption with keyword search," *The Computer Journal*, vol. 61, no. 8, pp. 1133–1142, 2018.

[18] Y. W. Hwang and I. Y. Lee, "A study on CP-ABE-based medical data sharing system with key abuse prevention and verifiable outsourcing in the IoMT environment," *Sensors*, vol. 20, no. 17, p. 4934, 2020.

[19] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, vol. 48, no. 177, pp. 203–209, 1987.

[20] C. Hu, J. Yu, X. Cheng, Z. Tian, and L. Sun, "CP-ABSC: An attribute based signcryption scheme to secure multicast communications in smart grids," *Mathematical Foundations of Computer Science*, vol. 1, no. 1, 2018.

[21] K. Emura, A. Miyaji, A. Nomura, K. Omote, and M. Soshi, "A ciphertext policy attribute-based encryption scheme with constant ciphertext length," in *International Conference on Information Security Practice and Experience (ISPEC)*, pp. 13–23, Xi'an, China, 2009.

[22] J. Kim, W. Susilo, F. Guo, M. H. Au, and S. Nepal, "An efficient KP-ABE with short ciphertexts in prime order groups under standard assumption," in *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, pp. 823–834, Abu Dhabi, United Arab Emirates, 2017.

[23] D. X. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," in *Proceeding 2000 IEEE symposium on security and privacy*, pp. 44–55, Berkeley, CA, USA, 2000.

[24] Y. W. Hwang and I. Y. Lee, "A study on data sharing system using ACPABE-SE in a cloud environment," *International Journal of Web and Grid Services*, vol. 17, no. 3, pp. 201–220, 2021.

[25] G. K. Verma, B. B. Singh, N. Kumar, and V. Chamola, "CB-CAS: certificate-based efficient signature scheme with compact aggregation for industrial internet of things environment," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2563–2572, 2019.

[26] K. A. Shim, "A new certificateless signature scheme provably secure in the standard model," *IEEE Systems Journal*, vol. 13, no. 2, pp. 1421–1430, 2018.

[27] X. Zhang, C. Jin, Z. Wen, Q. Shen, Y. Fang, and Z. Wu, "Attribute-based encryption without key escrow," in *International Conference on Cloud Computing and Security*, pp. 74–87, Nanjing, China, 2015.

[28] M. Chase, "Multi-authority attribute based encryption," in *Theory of cryptography conference*, pp. 515–534, Berlin, Germany, 2007.

[29] M. P. Hoyle and C. J. Mitchell, "On solutions to the key escrow problem," in *State of the Art in Applied Cryptography*, pp. 277–306, Leuven, Belgium, 1998.

[30] K. Zhang, J. Gong, S. Tang et al., "Practical and efficient attribute-based encryption with constant-size ciphertexts in outsourced verifiable computation," in *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, pp. 269–279, Xi'an, China, 2016.

[31] S. Belguith, N. Kaaniche, and G. Russello, "PU-ABE: Lightweight attribute-based encryption supporting access policy update for cloud assisted IoT," in *2018 IEEE 11th International Conference on Cloud Computing (CLOUD)*, pp. 924–927, San Francisco, CA, USA, 2018.

[32] J. Li, S. Cheng, Z. Cai, J. Yu, C. Wang, and Y. Li, "Approximate holistic aggregation in wireless sensor networks," *ACM Transactions on Sensor Networks*, vol. 13, no. 2, pp. 1–24, 2017.

[33] M. Chase and S. S. M. Chow, "Improving privacy and security in multiauthority attribute-based encryption," in *Proceedings of the 16th ACM conference on Computer and communications security*, pp. 121–130, Chicago Illinois, USA, 2009.

[34] Q. Li, H. Xiong, F. Zhang, and S. Zeng, "An expressive decentralizing kp-abe scheme with constant-size ciphertext," *International Journal of Network Security*, vol. 15, no. 3, pp. 161–170, 2013.

[35] B. Lynn, "The pairing-based cryptography (PBC) library," 2010, http://crypto.stanford.edu/pbc.

[36] L. Touati and Y. Challal, "Collaborative kp-abe for cloud-based internet of things applications," in *2016 IEEE International Conference on Communications (ICC)*, pp. 1–7, Kuala Lumpur, Malaysia, 2016.

[37] S. Y. Tan, K. W. Yeow, and S. O. Hwang, "Enhancement of a lightweight attribute-based encryption scheme for the Internet of Things," *IEEE Internet of Things Journal*, vol. 6, no. 4, pp. 6384–6395, 2019.

[38] B. Girgenti, P. Perazzo, C. Vallati, F. Righetti, G. Dini, and G. Anastasi, "On the feasibility of attribute-based encryption on constrained IoT devices for smart systems," in *2019 IEEE International Conference on Smart Computing (SMARTCOMP)*, pp. 225–232, Washington, DC, USA, 2019.