Attribute-Based Protocol for Secure Data Sharing Among Semi-Trusted Entities in Third-Party Service Environments

Jheng-Jia Huang, Guan-Yu Chen, Shao-Wei Tung*

Department of Information Management,
National Taiwan University of Science and Technology,
Taiwan
jhengjia.huang@mail.ntust.edu.tw, d11209103@mail.ntust.edu.tw, m10909101@mail.ntust.edu.tw

Abstract

Securing data storage and sharing has become a primary concern in today's application domains, prompting the exploration of Attribute-Based Encryption (ABE) protocols as a practical solution. ABE allows data owners to build access structures and share data with users based on specific attributes. However, existing ABE protocols typically require pairing operations, which leads to significant computational overhead. In addition, the inherent dependence of ABE systems on key generation centers raises security concerns. To address these challenges, we propose a novel data sharing protocol. Our protocol empowers data owners to retain control over data access, thereby reducing the risks associated with compromised key generation centers. This is achieved by fusing Shamir's secret sharing and attribute-based encryption schemes. In addition, we reduce the computational burden on the data owner by offloading part of the encryption overhead to the cloud. Furthermore, we confirm the robustness of our protocol against potential cryptographic attacks through formal proofs. These properties highlight the versatility of our solution to securely store and share data in various real-world scenarios.

Keywords: Cloud environment, Attribute-based protocol, Data sharing, Cryptographic protocols

1 Introduction

Traditionally, the confidentiality of transmitted and stored data has been safeguarded through the use of public key encryption, which provides robust security capabilities [1]. Public key encryption has typically served as a means for users to securely share data with other devices or users. While public-key encryption is well-suited for scenarios like client-server architectures [2-4], where the data owner precisely knows which user should receive the data, there are numerous situations where data owners may wish to share data based on attributes or policies associated with the receiving user, such as in the health care [5-7], wireless body area networks [8-10], smart city service [11-13] and Vehicular ad-hoc networks [14-16], and others. In these cases, public key encryption faces limitations. For instance, in network environments, the data owner may want to share data with a specific group of users. Using public key encryption would require encrypting the data multiple times with different public keys, leading to key storage issues. Additionally, if the data user only knows the conditions that the recipient must meet, without knowing the recipient's identity, public key encryption may not be effective. Moreover, access to the data may be restricted to users with specific credentials. Although trusted servers are commonly used to store and control data storage, compromising the server storing the data can result in the loss of data confidentiality, particularly in cloud environments. Therefore, this paper aims to discuss the limitations and challenges of cloud storage environments.

1.1 The Development and the Challenges of the Cloud Environment

As we delve deeper into the evolution of our technological era, we are uncovering the social advantages brought about by the cloud environment. The landscape of applications is growing in complexity, transcending the confines of a single physical location and expanding exponentially. From the era of floppy disks to ZIP drives, CDs, DVDs, and beyond, the evolution of storage media has been profound. Now, these traditional computing devices are being supplanted by non-local computing environments interconnected through the Internet — the cloud.

The inception of the precursor to the internet, Advanced Research Projects Agency Networks (ARPANET), by Larry Roberts and Bob Taylor in 1969, marked the beginning of a data computing revolution, further propelled by Licklider's visionary ideas. A plethora of cloud computing services, including Virtual Machines (VMs), virtual private networks (VPNs), Elastic Compute Cloud (EC2), Google Docs Services, and iCloud, have been developed by corporations over the past half-century, enhancing the functionality of the cloud environment.

With the burgeoning popularity of cloud computing, a myriad of information security challenges have surfaced [17]. Among these challenges, the security of virtual infrastructure stands out as a prominent concern. The inherent complexity of cloud environments renders it arduous to mitigate malicious attacks targeting cloud entities effectively. Furthermore, the privacy of cloud

*Corresponding Author: Guan-Yu Chen; Email: d11209103@mail.ntust.edu.tw
DOI: https://doi.org/10.70003/160792642026032702004

environments has become a focal point, particularly as many practical applications still transmit or store data without encryption. Even when the cloud service provider is deemed trustworthy, a breach in their security could compromise user privacy. Consequently, the provision of access to encrypted data and the implementation of robust access control mechanisms in cloud environments have become imperative, prompting the exploration of attribute-based encryption as a potential solution.

Sahai and Waters introduced a novel encryption method that enables data owners to encrypt plaintexts with predefined attributes and generate corresponding user keys prior to data transmission. Consequently, only users possessing attributes matching those predefined by the data owner can access the plaintext. Building upon this work, Goyal et al. termed this encryption method attribute-based encryption, highlighting its distinction from traditional encryption by enabling encryption of plaintexts for which keys have been generated. Crucially, with attribute-based encryption, data remains confidential even in untrusted storage server scenarios.

However, applying Sahai and Waters's scheme to cloud environments presents challenges, notably concerning the expressiveness of access control and the key management. While expressive attribute encryption schemes [18] have been proposed recently, there is the problem of excessive authority of a single entity. Unlike traditional public key encryption, where private keys are generated independently, in attribute-based encryption, the key generation center produces all user private keys using the master key. Although this centralization simplifies storage and certificate management, it poses a risk in cloud environments because due to the lack of a data protection scheme, the cloud service provider may have access to any data of the data owner.

Several attribute-based encryption schemes address too much authority for a single entity through two-party computing (2PC) protocols [19-23]. In 2016, Wang et al. proposed a property-based data sharing scheme for cloud computing, leveraging 2PC protocols to solve the key management problem. However, this scheme lacks a formal security certification. In 2013, Hur introduced a cryptographic protocol [24] tailored for cloud computing, achieving key escrow-free functionality but at the expense of increased computational burden on data owners. Similarly, in 2020, Varri et al. proposed an attribute-based data sharing scheme [23] enhancing security and efficiency but lacking formal security certification.

These existing solutions face three key limitations:

- High computational costs for data owners, particularly problematic for resource-constrained devices
- Lack of comprehensive security proofs against sophisticated attack models
- Insufficient mechanisms for controlling the authority of semi-trusted entities

Our work directly addresses these limitations through an innovative approach that redistributes computational overhead to cloud nodes while maintaining security, provides formal security proofs under both symmetric cryptography and attribute-based encryption models, and implements a refined trust model where no single entity, including the KGC, can independently access user data.

We define semi-trusted entities as those that will follow the protocol correctly but may attempt to learn unauthorized information. This differs from previous work where entities were often assumed to be either fully trusted or potentially malicious. Our protocol specifically limits the capabilities of semi-trusted entities by preventing the KGC from decrypting ciphertext without the symmetric key, ensuring cloud nodes can only access portions of symmetric keys, and requiring multiple entities to collaborate for legitimate data access. This refined trust model better aligns with real-world cloud environments where entities may be professionally reliable but not completely trustworthy, providing practical security without requiring fully trusted third parties.

1.2 Contributions

This paper tackles the excessive permissions for a single entity in cloud storage environments. Despite recent advances in the literature, researchers have yet to explore methods for reducing client costs while simultaneously proposing a formal security certification scheme. Leveraging the expressive attribute encryption scheme introduced by Waters [18], our paper builds upon this foundation to address the aforementioned challenges. The contributions of this paper are as follows:

- *Reduction of Computation Cost at the Client Side:* We observe that in the attribute encryption scheme proposed by Waters [18], user-side encryption involves pairing and exponential operations, imposing significant computational burdens on small sensing devices. Thus, we shift part of the computation to cloud nodes using symmetric encryption to alleviate the computation cost for data owners. As shown in Table 4, our scheme eliminates the need for pairing or exponential computations at the data uploader's side, effectively reducing computation costs.
- *Semi-Trusted Entities:* We utilize symmetric encryption and Shamir's secret sharing [25] to distribute decryption keys across various cloud nodes. As a result, although the key generation center retains the master key, it lacks the ability to decrypt ciphertext without the symmetric key. Furthermore, each cloud node exclusively holds a portion of the symmetric keys, preventing them from decrypting ciphertext independently. This approach effectively reduces the trust we need to give to third parties.
- *Addressing the Conspiracy Problem:* Previous research has discussed the conspiracy problem in the context of Shamir's secret sharing, where collusion among nodes could compromise the original secret. Our approach combines symmetric encryption and attribute-based encryption to mitigate this risk. Even in the event of collusion among nodes, decryption is only possible with the corresponding attribute key, effectively resolving the conspiracy problem.

- *Formal Security Proof:* We introduce two security models in Section 2.8. The first model pertains to symmetric cryptographic proofs, wherein security is maintained as long as the attacker cannot acquire the symmetric encryption key, ensuring resilience against chosen-ciphertext attacks (CCA). The second model centers on attribute-based encryption security, stipulating that decryption is unfeasible unless the attacker possesses a set of attributes that align with the access structure.

2 Preliminaries

Notation. Let Δ represent a set such that $|\Delta| \geq |t$, where $|\cdot|$ denotes the cardinality of the secret sharing set. Conversely, we denote Δ' as a set such that $|\Delta'| < t$.

2.1 Lagrange Polynomial Interpolation

Interpolation is a method used to find or determine the value of a polynomial function based on a known discrete set of data points. This technique finds numerous applications in mathematics and science, allowing for the creation of new data points within the range of a discrete data set of known points. The primary objective of interpolation is to compute unknown values based on the known data points.

Lagrange polynomial interpolation was initially published by Waring in 1779, rediscovered by Euler in 1783, and later formally presented by Lagrange in 1795. The formula [26] enables the construction of a polynomial function $P(x)$ with a degree of $t - 1$, utilizing t distinct points $((x_1, y_1), (x_2, y_2), \ldots, (x_t, y_t))$. Based on the observed data points, one can define the Lagrange basis polynomial as follows:

$$p_j(x) = \prod_{k \in \Delta, k \neq j}^{t} \frac{x - x_k}{x_j - x_k}$$

Subsequently, by inputting the data points $((x_1, y_1), (x_2, y_2), \ldots, (x_t, y_t))$, the function can be reconstructed as:

$$P(x) = \sum_{j \in \Delta}^{t} y_j p_j(x)$$

From the above formula, it is evident that in Lagrange polynomial interpolation, when $x_i = x_j$, there exists no $y_i \neq y_j$ in the polynomial. Only one unique corresponding value exists in the polynomial. In other words, if $P(x_i) = P(x_j)$, then x_i must equal x_j , indicating that these two points are identical.

2.2 Shamir Secret Sharing Scheme

Before 1979, there was no convenient and secure key management scheme to ensure the security of the key, so in 1979 Shamir proposed a threshold scheme [25] that could protect the security of the key and facilitate management by properly selecting the n and t parameters. Under this scheme, users can be granted rights by a majority decision, and at the same time, the scheme can also be used to defend against the threat of attackers. The following two stages will be included in the total in the (n, t) Shamir secret sharing scheme.

- Share Phase: In this phase the dealer will protect the secret s, so the dealer chooses the coefficients $a_1, a_2, \ldots, a_{t-1}$ from $[0, p)$, and sets the polynomial $q(x) = a_0 + a_1x + a_2x^2 + \ldots a_{t-1}x^{t-1}$. In the end the dealer computes shares $D_i = q(i)$ and sends the D_i to all the participants i. Reconstruction Phase: When a participant obtains t shares, use Lagrange polynomial interpolation to reconstruct function $q(x)$, then the participant can compute $s = q(0)$ *mod p*.

Proof. Given $t - 1$ shares we can obtain $t - 1$ equations

$$\begin{cases} D_1 = q(x_1) \bmod p \\ D_2 = q(x_2) \bmod p \\ \vdots \\ D_{t-1} = q(x_{t-1}) \bmod p \end{cases}$$

As a result, Lagrange polynomial interpolation allows anyone to determine the unknown data points based on the known data points. Every $s \in [1, p - 1]$ will have a (x_t, y_t) point, and if and only if one function of f can reconstruct the polynomial from these t points, and $s = f(0)$ *mod p*. In other words, the probability of all s is the same when the participant only has the $t - 1$ data points. □

2.3 Ciphertext-policy Attribute-based Encryption

Sahai and Waters (2006) introduced an attribute-based encryption scheme that enables specific access control policies. A notable characteristic of attribute-based encryption, in contrast to traditional encryption methods, is its capability to encrypt messages based on predefined attributes or policies. Essentially, attribute-based encryption allows for the encryption phase to precede the key generation phase. Consequently, when a user's attributes align with predefined criteria, the ciphertext can be decrypted using their private key. Additionally, Goyal et al. (2006) classify attribute-based encryption into two categories: ciphertext-policy attribute-based encryption (Bethencourt et al., 2007) and key-policy attribute-based encryption. This paper primarily focuses on ciphertext-policy attribute-based encryption.

Ciphertext-policy attribute-based encryption generally comprises four components:

- **Setup (1^λ).** The Setup algorithm takes security parameters 1^λ as input and produces public parameters along with a master secret key *MSK*.
- **Encrypt (*PK*, *M*, $\mathbb{A}$).** The encryption algorithm requires three input parameters: public parameters *PK* from the Setup phase, the message *M* to be encrypted, and the access structure $\mathbb{A}$. It outputs ciphertext *CT* such that only users possessing or matching the access structure can decrypt *CT* using their private key.
- **Key Generation (*MSK*, S).** The Key Generation algorithm takes the master secret key *MSK* from

the Setup phase and an attribute set S containing user attributes. It outputs the user's private key *SK*.

- **Decrypt (***PK*, *CT*, *SK***).** The Decrypt algorithm takes public parameters from the Setup phase, ciphertext *CT* from the Encrypt phase, and the user's private key *SK*. If the user's private key matches the access structure $\mathbb{A}$, the user can use this algorithm to decrypt the ciphertext *CT* and obtain the message *M*.

2.4 Ciphertext-policy Attribute-based Encryption

We consider $\mathbb{G}$ and $\mathbb{G}_{\mathbb{T}}$ as two multiplicative cyclic groups of prime order *p*. Let *g* be a generator of $\mathbb{G}$ and *e* be a bilinear map e: $\mathbb{G}\times\mathbb{G} \rightarrow \mathbb{G}_{\mathbb{T}}$. This bilinear map is characterized by the following properties:

1. Bilinearity: For all $g_1, g_2 \in \mathbb{G}$ and $x, y \in \mathbb{Z}_p$, we have $e\left(g_1^x, g_2^y\right) = e\left(g_1, g_2\right)^{xy}$.
2. Non-degeneracy: There exist $P, Q \in \mathbb{G}$ such that $e(P, Q) \neq 1$.
3. Computability: For any $P, Q \in \mathbb{G}$, $e(P, Q)$ can be computed efficiently.

We say *e* is a bilinear map if there exists $\mathbb{G}$ that can satisfy the above three properties.

2.5 Access Structure

Definition 1. **(*Access Structure* [27])** *Let* $\{P_1, P_2, \ldots, P_n\}$ *be a set of parties. A collection* $\mathbb{A} \subseteq 2^{\{P_1, P_2, \ldots, P_n\}}$ *is monotone if it has the property that for all B, C, if* $B \in \mathbb{A}$ *and* $B \subseteq C$, *then* $C \in \mathbb{A}$. *Furthermore, if the access structure* $\mathbb{A}$ *contains non-empty subsets* $2^{\{P_1, P_2, \ldots, P_n\}} \backslash \{\varnothing\}$, *then the elements in* $\mathbb{A}$ *are called authorized sets; otherwise, they are called unauthorized sets.*

2.6 Linear Secret Sharing Schemes (LSSS)

In secret sharing schemes, confidential information is distributed among multiple entities, each with a designated portion. The data owner establishes an access structure that determines how these portions can be combined to recover the secret. Two main access structures are tree-based and matrix-based. We focus on the matrix-based approach, common in linear secret sharing schemes [27]. This method involves share generation using a matrix. Next, we explore the details of the matrix-based access structure and linear secret sharing schemes. According to [27], a secret-sharing scheme $\prod$ over a set of entities *P* is "linear" (over $\mathbb{Z}_p$) if:

1. Shares designated to each entity form vectors over $\mathbb{Z}_p$.
2. A fundamental component of such schemes is the presence of a share-generating matrix $\mathbb{M}$, comprising ℓ rows and *n* columns, derived from the access structure $\mathbb{A}$. By assigning a label *i* to each matrix row through function ρ, a secret *s* is chosen for sharing, accompanied by random values $r_2, r_3, \ldots, r_n$ drawn from $\mathbb{Z}_p$. These values are then encapsulated within the column vector $\vec{v} = (s, r_2, r_3, \ldots, r_n)$. Application of the scheme $\prod$ to the matrix-vector product $\mathbb{M}\vec{v}$ yields ℓ shares of secret *s*, with each share being possessed by entity $\rho(i)$ and denoted as $(\mathbb{M}\vec{v})_i$.

Underlining the significance of linear reconstruction, as elucidated in [28], all linear secret sharing schemes are distinguished by the capability to linearly reconstruct secrets. Specifically, for an LSSS $\prod$ associated with access structure $\mathbb{A}$, any authorized set $S \in \mathbb{A}$ facilitates the formation of a subset $I \subset \{1, 2, \ldots, \ell\}$, defined as $I = \{i: \rho(i) \in S\}$. Given valid shares λ_i of a secret *s*, the existence of coefficients $\{w_i \in \mathbb{Z}_p\}_{i \in I}$ such that $\sum_{i \in I} w_i \lambda_i = s$ is ensured.

2.7 Decisional Parallel Bilinear Diffie-Hellman Exponent Assumption

Below is the refined definition of the decisional *q*-parallel bilinear Diffie-Hellman exponent problem as introduced by Waters in their publication [18]:

Let $\mathbb{G}$ be a multiplicative group with a chosen order *p* for security, and consider *g* as a generator of $\mathbb{G}$. Given $\vec{y} = (g, g^s, g^a, \ldots, g^{aq}, g^{aq+2}, \ldots, g^{2q})$, an adversary's task is to discern the value of $e = (g, g)^{aq+1s} \in \mathbb{G}_T$. If there exists an algorithm $\mathcal{B}$ capable of outputting $z \in 0,1$ and distinguishing the value of $e\ (g, g)^{aq+1s} \in \mathbb{G}_T$ using $\vec{y}$, then the algorithm $\mathcal{B}$ is considered to possess an advantage ϵ in solving the decisional *q*-parallel BDHE problem.

2.8 Security Definitions

Our protocol must establish security on two distinct fronts: the security of symmetric encryption for data confidentiality and the security of attribute-based encryption for access control. The first security model proves that our symmetric encryption scheme ensures message confidentiality even against active adversaries with chosen-ciphertext capabilities. The second model demonstrates that our attribute-based encryption component enforces proper access control, ensuring that only users with authorized attribute sets can access protected data. We formalize these requirements through the following security models:

Definition 2. *The game for INDistinguishability under Chosen-Ciphertext Attack* (*IND-CCA*) [29]

With a symmetric cryptosystem $\prod$, *a challenger* ψ *and an adversary* Γ *which engage in the following game.* ***Step* 1.** ψ *runs a setup algorithm. Params are the parameters* ψ *provides to* Γ. *The keys sk are provided to two oracles: one for encryption and another for decryption. Secure one-way hash functions like h are known as hash oracles. Oracles like the above hold the secret key secretly.* ***Step* 2.** *There is a sequence of encryption, decryption, and hash queries from the adversary. If the challenger receives an encryption query, denoted by* m^*, *it responds by returning* $\pi^* = E_{sk}(m^*)$ *to the adversary. As soon as an adversary issues a decryption query, denoted by* π^*, *the challenger responds with* $\rho^* = D_{sk}(\pi^*)$. *The challenger returns to the adversary* $\omega = h(\beta)$ *after receiving a hash query denoted by* β.

Challenge. (m_0, m_1) *is the plaintext pair output by the*

adversary. As soon as the challenger receives (m_0, m_1)*, the challenger randomly selects* $\theta \in \{0,1\}$ *and computes the ciphertext* $\pi = E_{sk}(m_0)$*. Then, the challenger returns* π *to the adversary.*

***Step* 3.** *An adversary issues the same encryption, decryption, and hashing queries as in Step*2*, but with the restriction that* $\pi^* \neq \pi$*.*

Guess. *Lastly, the adversary produces* $\theta' \in \{0,1\}$*. If* $\theta' = \theta$*, the adversary will win the game.*

The adversary is called an IND-CCA adversary with the guessing advantage $Adv_{\Pi}^{IND-CCA}(\Gamma) = \left| P_r[\theta = \theta'] - \frac{1}{2} \right|$*, since it participated in the game in polynomial time.*

Definition 3. *The Security Model for Ciphertext-Policy Attribute-Based Encryption* [30]

Setup. *By running the Setup algorithm, the challenger gives the adversary the public parameters, PK.*

***Phase* 1.** *Repeated private keys are made by the adversary corresponding to set attributes* $S_1, \ldots, S_{q_1}$*.*

Challenge. M_0 *and* M_1 *are equal messages sent by the adversary. Furthermore, there is no set* $S_1, \ldots, S_{q_1}$ *from Phase* 1 *that can match the adversary's challenge access structure* $\mathbb{A}^*$*. Flipping a coin, the challenger encrypts a random value* M_b *under the access structure* $\mathbb{A}^*$*. The ciphertext* CT^* *is given to the adversary.*

***Phase* 2.** *Phase* 1 *is repeated with the restriction that none of the sets of attributes* $S_{q_{1+1}}, \ldots, S_q$ *satisfy the access structure corresponding to the challenge.*

Guess. *The adversary outputs the guess* b' *of* b*.*

Finally, the advantage of an adversary $\mathcal{A}$ *in this game is defined as* $P_r[b' = b] - \frac{1}{2}$ *. In the paper* [30]*, a decryption query can be performed in both Phases* 1 *and* 2 *of the security model for handling chosen-ciphertext attacks the authors note.*

3 Attribute-Based Protocol for Secure Data Sharing Among Semi-Trusted Entities in Third-party Service Environments

In this section, we introduce a novel attribute-based protocol designed to reduce reliance on third-party entities in cloud environments. Our framework includes five key entities - Key Generation Center (KGC), data owners, users, data centers, and cloud service providers. The KGC generates individual user private keys but notably cannot decrypt the secrets stored in the database independently. Data owners determine what information to share and specify required attributes for access, while cloud service providers store encrypted data and key shares. Users can only access shared data when their attributes meet the designated criteria and they receive proper authorization from the cloud service provider.

Our framework operates under two key assumptions. First, we assume that collusion between the data center and other cloud service providers is not feasible. Unlike traditional setups where servers are fully trusted, we recognize potential risks during data transmission. To mitigate these risks, we employ message encryption and distribute key shares among multiple providers, effectively reducing the cloud server trust level to semi-trusted. Second, users obtaining shares from different providers must be authenticated members, with cloud service providers responsible for preventing malicious access attempts.

Table 1 outlines the notation used in our framework. The data owner decides what information to share and specifies the attributes required for user access, while the KGC generates individual user private keys. Crucially, the KGC cannot decrypt the secrets stored in the database alone. By leveraging a combination of attribute-based encryption and symmetric encryption techniques, only users meeting the designated access criteria and authorized by the cloud service provider can access shared data. Figure 1 depicts our comprehensive framework.

Table 1. Notations of the proposed scheme

Notation	Meaning
p, p_1, q_1	The large prime numbers used as security parameters
PK_{dc}	The public key of data center for encryption
SK_{dc}	The private key of data center for decryption
$\mathcal{U}$	The universe of possible attributes in polynomial bound
$\mathbb{A}$	The access structure defining required attribute combinations
$H(\cdot)$	A secure one-way cryptographic hash function
r	A random secret value chosen by the data owner to encrypt their message, acting as an encryption key
C_i	The i^{th} cloud service providers
D_i	The i^{th} share of secret r stored in C_i
n	The number of participating cloud service providers
t	The threshold for secret reconstruction ($t \leq n$)
M	The message of the data owner
T	The timestamp for preventing replay attacks
U_i	The user who want to get the message
$E_r(\cdot)$	The symmetric encryption function using secret r
$D_r(\cdot)$	The symmetric decryption function using secret r

3.1 Setup Phase

During the setup phase, the KGC establishes the public parameters and functions necessary for our proposed construction. Specifically, the KGC selects three large primes p, p_1, q_1, and two multiplicative groups $\mathbb{G}$, $\mathbb{G}_{\mathbb{T}}$ with a prime order p. Let g denote a generator of $\mathbb{G}$, and e: $\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_{\mathbb{T}}$ represent a bilinear map.

Subsequently, the KGC defines the universe as $\mathcal{U} = 1, 2, \ldots, k$ and randomly selects $h_1, h_2, \ldots, h_k \leftarrow \mathbb{G}$, alongside choosing α, $a \leftarrow Z_p$. Then, the KGC computes $Y = e(g, g)^{\alpha}$ and $h = g^a$, while also selecting a secure one-way hash function $H(\cdot)$: $0,1^* \rightarrow Z_p$. The system parameters PK are then specified as $(g, h, Y, h_{i\,i=1}^{k}, H(\cdot))$, with $msk = g^a$ securely stored.

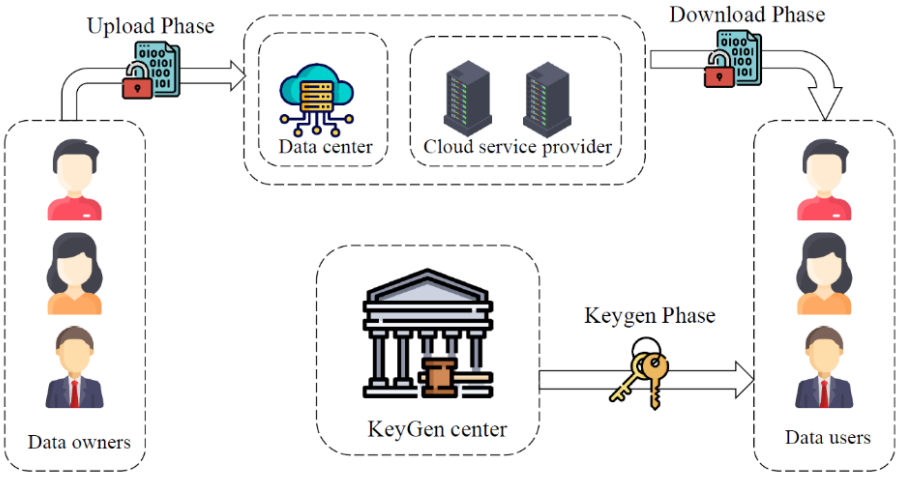

Figure 1. The flow of our scheme

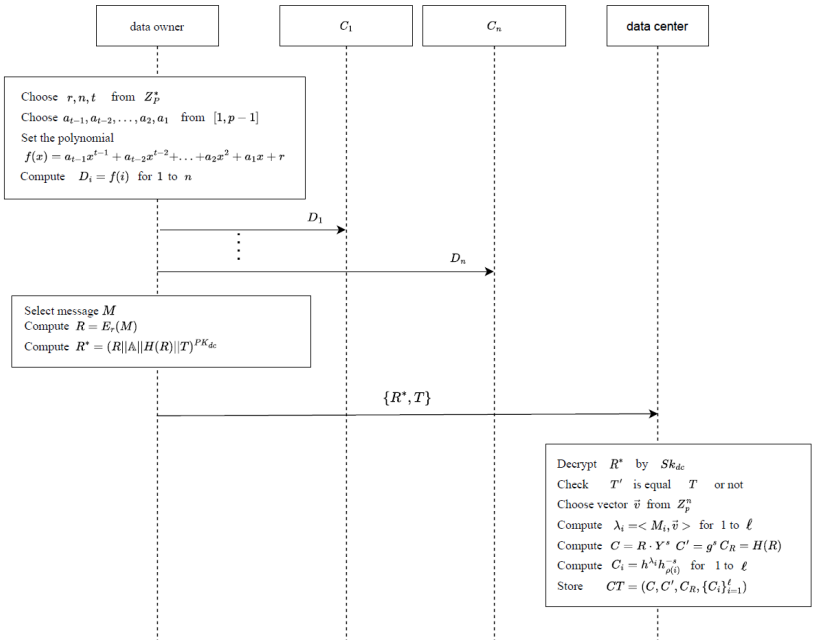

Figure 2. The diagram of upload phase

Finally, the data center employs the system parameters PK to calculate $N = p_1 \cdot q_1$, and chooses $\hat{e}$ such that $gcd(\hat{e}, \phi(N)) = 1$. It then computes $d = (\hat{e})^{-1} \bmod \phi(N)$ and generates $PK_{dc}=(N, \hat{e})$ as its public key, with $SK_{dc} = d$ serving as the private key for the data center. It is pertinent to note that any asymmetric cryptography method can be employed; herein, we illustrate the utilization of RSA as an exemplar.

3.2 Upload Phase

In scenarios such as IoT, medical, and Industry 4.0, where data access is contingent upon adherence to specified access structures, the data sharing process between the data owner and the data center unfolds through the following steps (Figure 2):

Step 1: Share Distribution ($U_i \rightarrow C_i : D_i$)

Initiating the process, the data owner selects a random number r from Z_p^*. Subsequently, utilizing Shamir secret sharing, the data owner determines (n, t), where n denotes the total number of shares to be distributed and t represents the threshold number of shares required for secret reconstruction. The data owner then selects random coefficients a_{t-1}, a_{t-2}, ..., a_2, a_1 to define a polynomial function $f(x) = r + a_1x$ .... $a_{t-1}x^{t-1}$. Consequently, $D_1 = f(1)$, $D_2 = f(2)$ …, $D_n = f(n)$ are computed and transmitted to distinct cloud service providers.

Step 2: Data Center Communication ($U_i \rightarrow DC$: R^*, T)

Upon dispatching the shares to the cloud service provider, the data owner proceeds to encrypt the message M using the previously chosen random number r to generate $R = E_r(M)$. Subsequently, the data owner encrypts R,

along with the access structure $\mathbb{A}$, $H(R)$, and timestamp T, using the data center's public key PK_{dc}, resulting in $R^* = (R||\mathbb{A}||H(R)||T)^{PKdc}$. This parameter R^*, along with the timestamp T, is then forwarded to the data center.

Upon receipt,

1. Decrypts R^* using SK_{dc} to extract R, $\mathbb{A}$, T', and $H(R)$
2. Verifies $|T'-T| \leq \Delta T$, where ΔT represents the acceptable time delay
3. Encrypts value R according to access structure $\mathbb{A} = (\mathrm{M}, \rho)$, with M being an $\ell \times n$ matrix

Having encrypted the value R, the data center randomly selects a vector $\vec{v} = (s, y_2, ..., y_n)$ from $\mathbb{Z}p^n$. For all $i = 1$ to ℓ, the data center computes $\lambda_i = <M_i, v>$, $C_i = h^{\lambda i} h^{-s} p(i)$, $C = R \cdot Y^s$, $C' = g^s$, $C_R = H(R)$. Finally, the data center stores the $CT = (C, C', C_i$ for $i = 1, \ldots, \ell, C_R)$ in its database.

3.3 User Keygen Phase

During the user keygen phase, we assume each user must register with the KGC. This phase involves the user registering with the KGC and performing the following steps (Figure 3):

The user keygen phase consists of two main steps:

Step 1: $U_i \rightarrow KGC$: id_i, S

Initially, U_i will send id_i and attribute set S to KGC. When KGC receives the message, the KGC will check if id_i is duplicated, if not KGC will select a random number t from Z_p, then compute $K = g^{\alpha} h^t$, $L = g^t$ and calculate $K_x = h_x^t$ for all $x \in S$. Finally, KGC generates $\mathrm{D} = (K, L, \{K_x\}_{x \in S})$ and sends D to $User_i$.

Step 2: $KGC \rightarrow U_i$: D

After receiving their private key D, U_i will store the private key independently.

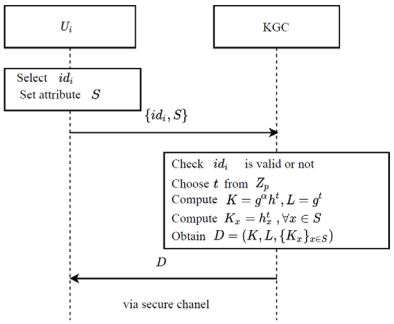

Figure 3. The diagram of user keygen phase

3.4 Download Phase

Any user can request interest data from the data center during the download phase. Assuming that a user wants to get interest data, they will perform the following steps (Figure 4):

Step 1: Request Initiation ($U_i \rightarrow DC$: *request*)

Upon receiving the request from the U_i, the data center will sign the CT in the database and timestamp with its private key to get $CT^* = (CT||T)^{SKdc}$, and then send $\{CT^*, T\}$ to U_i.

Step 2: Response Verification ($DC \rightarrow U_i$: CT^*, T)

The message CT^* will be verified when U_i receives it by using the data center public key to get CT^*, T', and the U_i will check whether $|T'-T| \leq \Delta T$ where ΔT is the time delay between device communication. If the time delay is acceptable, U_i will store CT and send an individual request to the different cloud service providers.

Step 3: Share Collection and Decryption ($C_i \rightarrow U_i$: D_i)

1. Compute Lagrange coefficients:
$$L_j(x) = \prod_{k \in \Delta, k \neq j}^{t} \frac{x - x_k}{x_j - x_k}$$
2. Reconstruct the function: $f(x) = \sum_{i \in \Delta} y_i L_i(x)$
3. Obtain random number r by computing $f(0)$
4. Use private key $\mathrm{D} = (K, L, \{K_x\}_{x \in S})$ to decrypt
$$CT' = \left(C, C', \{C_i\}_{i=1}^{\ell}, C_R\right)$$
5. Compute $\Omega = \dfrac{e(C', K)}{\prod_{i \in I}\left(e(C_i, L) e\left(C', K_{p(i)}\right)\right)^{\omega_i}}$ i
6. Calculate R' through $\dfrac{C}{\Omega}$
7. Verify integrity by checking if $C'_R = H(R')$ equals C_R
8. Finally, recover message $M = D_r(R')$

4 Security Proof

We assume the data center cannot conspire with the cloud service provider. Thus, the security proof has two parts: for the cloud service provider, based on the bilinear Diffie-Hellman index, and for data centers, based on symmetric encryption. Our architecture ensures each attribute in the access matrix appears only once. Using an encoding technique inspired by [30], attributes like "Student" become "Student1," "Student2," etc. This increases the user's private key size by k_{max}, the maximum number of attributes in M. Our construction is secure if $\rho(\cdot)$ is injective, though encoding may cause attributes to appear k_{max} times in the user's secret key. We delegate computing tasks to cloud services to help IoT devices handle attribute-based encryption, enabling fine-grained access control in cloud environments. Detailed security proofs follow.

Theorem 1. *The proposed key escrow protocol for the upload phase is secure based on the IND-CCA security under the decisional parallel bilinear Diffie-Hellman exponent assumption.*

Lemma 1. *The proposed protocol in the upload phase constitutes a secure key escrow protocol for both the data owner and the data center under the assumption of the IND-CCA security of the decisional parallel bilinear Diffie-Hellman exponent problem.*

Proof. Let $\mathcal{A}$ be a polynomial-time adversary with a non-negligible ϵ guessing advantage in playing the game as defined in **Definition 3** (Figure 5). Let simulator Γ simulate the data center where the decisional q-BDHE problem is played, given a challenging matrix M^* of dimensions at most q columns.

Init Oracle 1. The simulator takes parameters

$\vec{y} = \left(g, g^{s}, g^{a}, ..., g^{a^{q}}, g^{a^{q+2}}, ..., g^{a^{2q}}\right)$ and T as inputs. It receives the adversary's challenge access structure $\mathbb{A}^{*} = (M^{*}, \rho^{*})$, where M^{*} has $n^{*} \leq q$ columns.

Setup Oracle 1. The simulator selects two multiplicative groups $\mathbb{G}$ and $\mathbb{G}_{\mathbb{T}}$ with order p, with g as a generator and e as a bilinear map. It publicly defines a one-way hash function $H(\cdot)$.

The simulator picks large primes p_1 and q_1, generating the adversary's keys PK and SK. It chooses $\alpha' \in \mathbb{Z}_p$ and sets $\alpha = \alpha' + a^{q+1}$, then defines h_x values based on M^{*} and ρ^{*}.

Phase 1.

Upload Oracle 1. The adversary provides r and plaintext M. The simulator chooses random values and defines polynomial $f(x) = a_{t-1}x^{t-1} + ... + a_1x + r$. Shares $D_x = f(x)$ are computed and returned. The plaintext M is encrypted with symmetric encryption to get R. The attribute encryption is based on q-BDHE assumptions, producing ciphertext $CT = \{C, C', C_R, C_i\}$.

KeyGen Oracle 1. For an attribute set S not satisfying M^{*}, a vector $\vec{w}$ satisfying $\vec{w} \cdot M^{*} = 0$ and $\vec{w} \cdot (1, 0, ..., 0) = -1$ is found. The simulator computes K and K_x based on $\vec{w}$ and the attribute set S.

Download Oracle 1. If CT matches CT^{*}, the query terminates. Otherwise, the simulator decrypts CT using a user's private key if it satisfies M^{*}, revealing plaintext M.

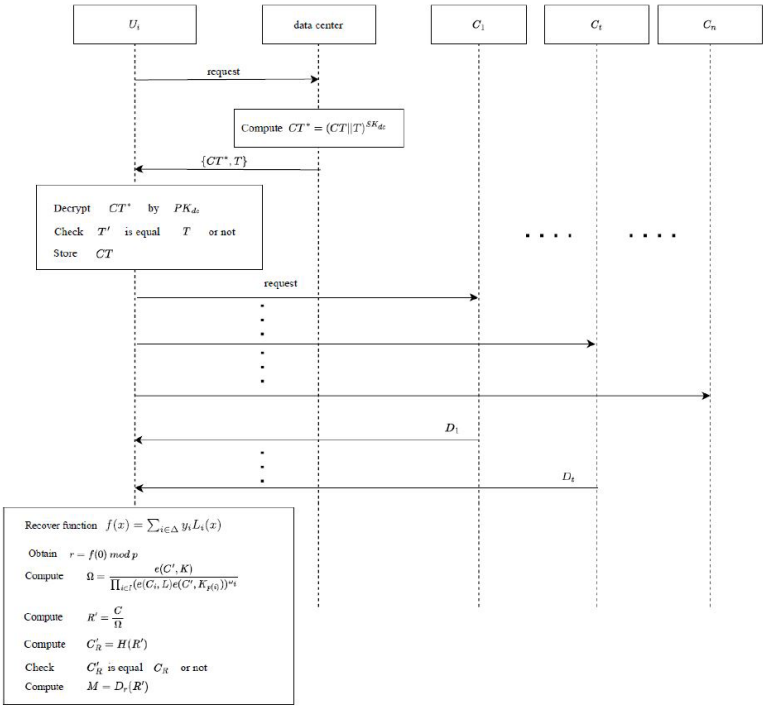

Figure 4. The diagram of download phase

Challenge Oracle 1. The adversary provides messages M_0, M_1 and a random value r. The simulator flips a coin β, encrypts M_β, and produces challenge ciphertext CT^{*}. The adversary guesses whether CT^{*} is encrypted with M_0.

Phase 2.

Guess 1. If the adversary's guess is correct, Γ outputs 0; otherwise, it outputs 1. The probability of winning is:

$$Pr\left[\Gamma\left(\vec{y}, T = e(g,g)^{a^{q+1}s}\right) = 0\right] = \frac{1}{2} + Adv_A$$

Thus, an adversary solving the hard problem with non-negligible advantage implies the simulator has the same advantage in solving the decisional q-BDHE problem.

□

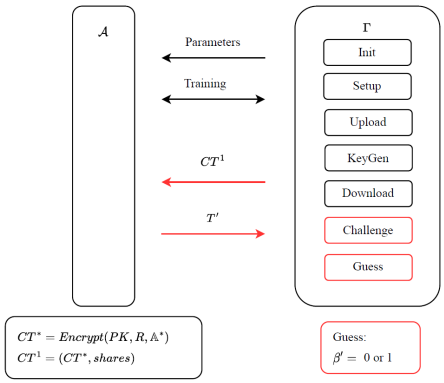

Figure 5. Security proof for lemma 5.1

Theorem 2. *The proposed key escrow protocol for the upload phase is securely based on the IND-CCA security of the underlying symmetric cryptosystem.*

Lemma 2. *The proposed protocol in the upload phase is a secure key escrow protocol for the KGC and data center under the assumption of the IND-CCA security of the underlying symmetric cryptosystem.*

Proof. Suppose that a polynomial-time adversary $\mathcal{A}$ with a non-negligible guessing advantage ϵ is playing the game defined in 2 and has targeted a secret value r^*, claiming it to the simulator Γ (Figure 6), which simulates the data center.

Setup Oracle 2. The simulator selects two multiplicative groups $\mathbb{G}$ and $\mathbb{G}_T$ with order p, with g as a generator and e as a bilinear map. It publicly defines a secure one-way hash function $H(\cdot)$ and creates the public key PK and private key SK using large primes p_1 and q_1. Random values α, $a \in \mathbb{Z}_p$ are chosen, and parameters $h_1, h_2, \ldots, h_U$ are defined from the attribute universe $\mathcal{U}$.

Phase 3.

Upload Oracle 2. The adversary queries $(r, M, \mathbb{A}^* = (M^*, \rho^*))$. The simulator chooses random values n, $t \in \mathbb{Z}_p$ and coefficients a_i, sets the polynomial $f(x)$, and computes shares $D_x = f(x)$, returning them to the adversary.

The message M is encrypted with r to get R. The simulator creates attribute-based encryption as follows: $C' = g^s$, $C_R = H(R)$, $C = R \cdot e(g, g)^{\alpha s}$. For C_i, based on LSSS, the simulator chooses random values and computes $C_i = g^{a\lambda_i} h_{\rho^*(i)}^{-s}$. The ciphertext $CT = \{C, C', C_R, C_i\}$ is returned.

KeyGen Oracle 2. The adversary queries private keys. For an attribute set S, the simulator chooses $t \in \mathbb{Z}_p$ and computes $L = g^t$, $K = g^{\alpha} g^{at}$. For all $x \in S$, K_x is computed as h_x^t. The private key $D = (L, K, \{K_x\}_{x \in S})$ is returned.

Download Oracle 2. If CT uses r^*, the query is aborted. Otherwise, if a user u with attributes S_u satisfies CT, the simulator retrieves u's private key, decrypts CT to get R, reconstructs r, and decrypts R to get M. The result is returned to the adversary.

Challenge Oracle 2. The adversary provides messages M_0, M_1 and access structure $\mathbb{A}^*$. The simulator flips a coin β, encrypts M_β to get R^*, and creates challenge ciphertext CT^*. The adversary receives shares of r^* and CT^*, which is encrypted with M_0 or not.

Phase 4.

Guess 2. If the adversary provides shares that reconstruct the decryption key and $M^*_0 = M_0$, the simulator outputs 0; otherwise, it outputs 1. The probability of winning is:

$$Pr\left[\Gamma\left(M_\beta = M_0\right) = 0\right] = \frac{1}{2} + Adv_A$$

Thus, if an adversary solves the hard problem with non-negligible advantage, the simulator has the same advantage in the game.

□

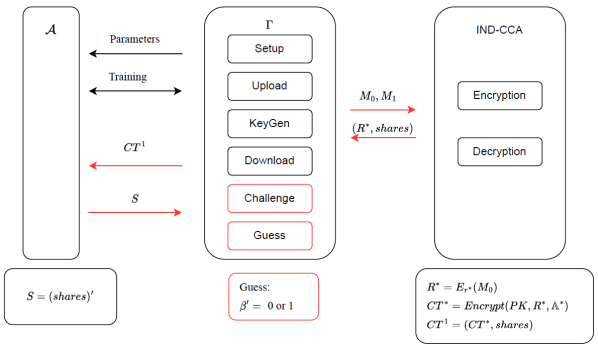

Figure 6. Security proof for lemma 5.2

5 Performance Comparison and Results

5.1 Property Comparison

In this section, we compare the properties of attribute-based encryption schemes including those proposed by [22], [23], and [24] based on their security proof, access structure, and key escrow resistance. Low computational cost indicates that the data owner does not incur significant expenses to encrypt the file or message before uploading it to the cloud. Security proof ensures that the scheme has been theoretically proven to be secure based on the security model discussed in the preliminaries section. Access structure defines the rules for the attributes that users must possess to decrypt the ciphertext. Key escrow resistance refers to a solution in attribute-based encryption that prevents ciphertext from being compromised by a fully trusted server. Two semi-trusted servers mean that the environment requires two semi-trusted servers to handle key generation. The comparison is summarized in Table 2.

5.2 Communication Comparison

Our assumptions in this section are a secure one-way hashing function's output size which is 256 bits, an asymmetric cryptosystem output size which is 1024 bits, a symmetric cryptosystem output size which is 128bits, the size of a random value is 64bits, the size of a timestamp is 64bits, the $|\mathbb{G}|$ is the size of an element in $\mathbb{G}$, the $|\mathbb{G}_T|$ is the size of an element in $\mathbb{G}_T$, and the $|\mathbb{G}_\tau|$ is the size of the access tree.

Table 2. Properties comparison

Properties	[22]	[23]	[24]	Ours
Key escrow free	Y	Y	Y	Y
With two semi-trusted server	Y	Y	Y	Y
Low computational cost	N	N	N	Y
Access structure	Tree	LSSS	Tree	LSSS
Security model	N	ROM	N	STD
The security Level	N	CKA	N	CCA

The ROM is the random oracle; the STD is the standard model
The CKA proposed in [23] is the chosen keyword attack

As a result of the comparison, in our proposed construction, the user private key is relatively short because Shamir secret sharing is used to solve the key escrow problem. A scheme proposed Wang et al. assigns different weights to each attribute in the access tree, and for each attribute, individual elements are generated in the ciphertext to allow for decryption. Thus, it leads to $\sum_{i=1}^{\ell}\left(w_i - w_{i_1} + 1\right)|\mathbb{G}| + |\mathbb{G}_T| + |\mathbb{G}_\tau|$ bits for the size of the ciphertext and $(k + 2)|\mathbb{G}|$ bits for the size of the user private key. A scheme proposed by Varri et al. provides users with the ability to perform a keyword search for ciphertext by attribute-based encryption and, simultaneously, uses the 2PC protocol to allow users to prevent key escrow problems. Thus, it leads to $(2\ell + 2)\ |\mathbb{G}| + |\mathbb{G}_T|$ for the size of the ciphertext and $(3+ k)|\mathbb{G}|$ for the size of the user private key. In the scheme proposed by Hur, the key escrow problem is solved using the 2PC protocol, and user revocation is handled via a proxy re-encryption mechanism. Thus, it leads to $(2\ell + 1)\ |\mathbb{G}| + |\mathbb{G}_T| + |\mathbb{G}_\tau|$ for the size of the ciphertext and $(2k + 2)|\mathbb{G}|$ for the size of the user private key. Table 3 shows the user private key size and ciphertext size for each related work and our proposed scheme, which we calculated as part of our analysis.

Table 3. The communication cost of the ciphertext and private key

Scheme	Size of *CT*	Size of private key
[22]	$\sum_{i=1}^{\ell}\left(w_i - w_{i_1} + 1\right)\|\mathbb{G}\| + \|\mathbb{G}_T\| + \|\mathbb{G}_\tau\|$	$(k + 2)\|\mathbb{G}\|$
[23]	$(2\ell + 2)\ \|\mathbb{G}\| + \|\mathbb{G}_T\|$	$(k + 3)\|\mathbb{G}\|$
[24]	$(2\ell + 1)\ \|\mathbb{G}\| + \|\mathbb{G}_T\| + \|\mathbb{G}_\tau\|$	$(2k + 2)\|\mathbb{G}\|$
Ours	$(\ell + 2)\ \|\mathbb{G}\| + \|\mathbb{G}_T\|$	$(k + 2)\|\mathbb{G}\|$

ℓ: the number of attributes in the access structure
k: the number of attributes associated with the private key of the user
w_i: Maximum weight of attribute i in the system proposed by [22]
w_{i1}: Weight of attribute i in the ciphertext *CT* proposed by [22]
$|\mathbb{G}|$: The size of an element in $\mathbb{G}$
$|\mathbb{G}_T|$: The size of an element in $\mathbb{G}_T$
$|\mathbb{G}_\tau|$: The size of an access structure

5.3 Computation Comparison

This section compares the computation costs of attribute-based encryption for both the data owner and the user sides with those in related works. Table 4 defines various parameters representing different computation costs across all phases. Here, T_H represents the time taken for one-way secure hash operation, T_{AES} denotes the time taken for 128-bit AES symmetric encryption or decryption operation, T_{RSA} signifies the time taken for 1024-bit RSA asymmetric encryption or decryption operation, T_{Poly} indicates the time taken for Lagrange polynomial interpolation operation, $T_{\mathbb{G}}$ and $T_{\mathbb{G}_T}$ represent the time taken for exponentiation operations in groups $\mathbb{G}$ and $\mathbb{G}_T$ respectively, T_e stands for the cost of pairing, and T_m represents the cost of modular multiplication. It's referenced that $T_H \approx T_{AES} \approx 0.4T_m$ based on [31-32], $T_{RSA} \approx 240T_m$ based on [33], and $T_p \approx 5T_e$, $T_e \approx 240T_m$, $T_{\mathbb{G}} \approx T_{\mathbb{G}_T} \approx 29T_m$ based on [34].

Table 4. The computation cost of the data owner side and user side

Scheme	The data owner side	The user side
[22]	$\left\{\left[\sum_{i=1}^{\ell}\left(w_i - w_{i_1} + 2\right)\right] + 1\right\}T_{\mathbb{G}} + 2T_{\mathbb{G}T}$	$(2k + 1)T_p + (2\log\ell+2)T_{\mathbb{G}_T}$
[23]	$(2\ell + n_k + 5)T_{\mathbb{G}} + (n_k + \ell)T_H + T_{\mathbb{G}T}$	$(d + 4)T_{\mathbb{G}} + (2k + 1)T_p + dT_H$
[24]	$(2\ell + 1)T_{\mathbb{G}} + T_{\mathbb{G}_T}$	$(2k + 2)T_p + mkT_{\mathbb{G}} + \log\ell T_{\mathbb{G}_T}$
Ours	$T_{AES} + T_{RSA} + T_{Poly} + T_H$	$(2k + 1)T_p + kT_{\mathbb{G}_T} + T_{\mathbb{G}} + T_{Poly} + T_H + T_{AES}$

ℓ: the number of attributes in the access structure
k: the number of attributes associated with the private key of the user
w_i: Maximum weight of attribute i in the system proposed by [22]
w_{i1}: Weight of attribute i in the ciphertext *CT* proposed by [22]
n_k: the number of keywords in the index proposed by [23]
d: the number of keywords in the query proposed by [23]
m: the number of users in an attribute group [24]
$T_{\mathbb{G}}$: the cost of exponentiation in $\mathbb{G}$
$T_{\mathbb{G}_T}$: the cost of exponentiation in $\mathbb{G}_T$
T_p: the computation time of a pairing
T_H: the computation time of the hash algorithm
T_{AES}: the computation time of 256-bit AES encryption/decryption
T_{Poly}: the computation time of the polynomial function
T_e: the cost of a modular exponentiation
T_m: the cost of a modular multiplication

Our proposed construction resolves the key escrow problem from the Key Generation Center (KGC), which can act as a semi-trusted server. Additionally, our proposed construction protects against malicious users through the shares of the random number r. The scheme by Wang et al. employs the weight of the tree-based access structure for data encryption and recursive methods for ciphertext decryption. Consequently, it incurs a computation cost of $\left\{\left[\sum_{i=1}^{\ell}\left(w_i - w_{i_1} + 2\right)\right] + 1\right\}T_{\mathbb{G}} + 2T_{\mathbb{G}_T}$ for the data owner side and $(2k + 1)T_p + (2\log\ell+2)T_{\mathbb{G}_T}$ for the user side. The scheme introduced by Varri et al. allows users to search for keywords while addressing key escrow issues using Two-Party Computation (2PC). Consequently, it incurs a computation cost of $(n_k + 3)T_{\mathbb{G}} + (n_k + \ell)T_H + T_{\mathbb{G}T} + (2\ell + 2)T_{\mathbb{G}}$ for the data owner side and $(d + 4)T_{\mathbb{G}} + (2k + 1)T_p + dT_H$ for the user side. The scheme by Hur utilizes the tree-based access structure for plaintext encryption and data re-encryption for immediate user revocation. Consequently, it incurs a computation cost of $(2\ell + 1)T_{\mathbb{G}} + T_{\mathbb{G}_T}$ for the data owner side and $(2k + 2)T_p + mkT_{\mathbb{G}} + \log\ell T_{\mathbb{G}_T}$ for the user side. In our analysis, we systematically calculated the computation costs of each related work and the proposed scheme, as shown in Table 4.

6 Implementation and Simulation

In this section, we present the results of implementing our protocol with a focus on both theoretical correctness and practical performance. Our implementation covers the complete protocol workflow, including the Shamir secret sharing scheme and the ciphertext policy attribute-based encryption (CP-ABE) components. To demonstrate practical applicability, we evaluate our protocol under different access structures and threshold configurations.

Access structures significantly impact the computational costs of both encryption and decryption operations. Table 5 details the different levels of access structures used in our evaluation, ranging from simple single-attribute structures to complex multi-attribute combinations with AND/OR operations.

Table 5. The detail of access structure

Level	The access structure
1	(ONE)
2	(ONE and TWO)
3	((ONE and THREE) and (TWO OR FOUR))
4	(((ONE or TWO) and (THREE or ELEVEN)) and ((FOUR and FIVE) or (SEVEN and EIGHT)))
5	(((ONE or TEN) and (TWO or EIGHT) and (SIX or SEVEN)) and (THREE or NINE) and ((FOUR or TWELVE) and (FIVE or ELEVEN)))

In our architecture, the thresholds of different secret sharing schemes do not impact the execution time of the encryption phase. Thus, in Figure 7, we present the required encryption time for our architecture under different access structures. It's evident that the required encryption time grows approximately exponentially as the access structures become more complex.

The threshold value, represented as (n, t), significantly impacts decryption performance, where n represents the total number of shares and t represents the minimum shares required for secret reconstruction. Figure 8 demonstrates how different threshold configurations affect the overall computation time. Our results show that while higher threshold values increase computational overhead, the system maintains acceptable performance for practical security requirements.

To further analyze the impact of threshold values, we conducted experiments with fixed total shares n while gradually increasing the threshold t. As shown in Figure 9, the incremental increase in threshold values does not dramatically impact the system's computational efficiency, suggesting our protocol remains practical even with higher security requirements.

These experimental results validate that our protocol achieves a practical balance between security and performance. The implementation demonstrates that even with complex access structures and high threshold values, the computational overhead remains within acceptable bounds for real-world applications.

For the simulations, we utilized the hosting system available at https://github.com/sagrawal87/ABE. The simulation environment is detailed in Table 6.

Table 6. Simulation environment

OS	64bit Ubuntu 20.04
CPU	Intel I5-8400 at 2.8GHz
RAM	32GB
Runtime	Python 3.8

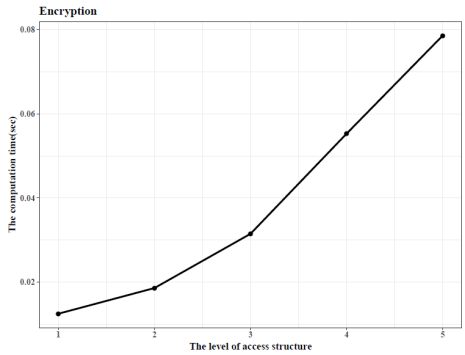

Figure 7. The encryption time of our scheme

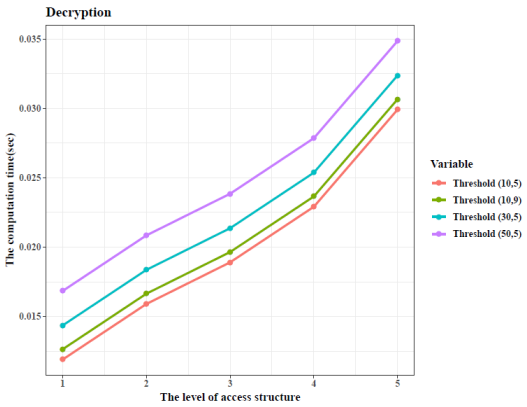

Figure 8. The decryption time under different thresholds

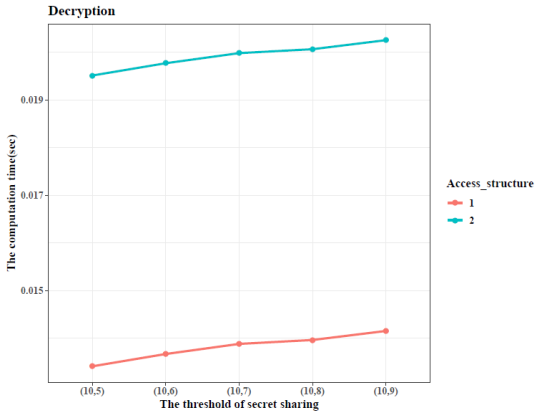

Figure 9. The decryption time under different access structures

7 Conclusion and Future Work

Our protocol enhances data privacy in attribute-based encryption by reducing trust requirements for cloud providers and key generation centers while preventing collusion. Security proofs and performance evaluations

demonstrate improved efficiency in computation and communication compared to existing approaches.

Implementation requires careful consideration of cloud infrastructure integration, key management systems, and performance optimization, particularly for IoT deployments. Network stability and scalability remain critical factors as user bases expand.

For future work, we plan to conduct comprehensive simulations with various cloud service providers to validate real-world performance. We will work on removing the requirement for consumers to maintain cloud service provider membership and develop more efficient authentication mechanisms.

Acknowledgments

This work was partially supported by the National Science and Technology Council of Taiwan, under grants NSTC 112-2221-E-011 -092 -MY2, NSTC 112-2634-F-011 -002 -MBK, NSTC 112-2221-E-011 -094 -MY2, and NSTC 112-2218-E-011 -008 -.

References

[1] J. Nechvatal, *Public-key cryptography*, National Institute of Standards and Technology, Report Number 800-2, April, 1991.

[2] R. Chen, Y. Mu, G. Yang, F. Guo, X. Huang, X. Wang, Y. Wang, Server-aided public key encryption with keyword search, *IEEE Transactions on Information Forensics and Security*, Vol. 11, No. 12, pp. 2833–2842, December, 2016.

[3] P. Xu, H. Jin, Q. Wu, W. Wang, Public-key encryption with fuzzy keyword search: A provably secure scheme under keyword guessing attack, *IEEE Transactions on Computers*, Vol. 62, No. 11, pp. 2266–2277, November, 2013.

[4] D. Boneh, G. D. Crescenzo, R. Ostrovsky, G. Persiano, Public key encryption with keyword search, *International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, Interlaken, Switzerland, 2004, pp. 506–522. https://doi.org/10.1007/978-3-540-24676-3_30

[5] S. Alshehri, S. P. Radziszowski, R. K. Raj, Secure access for healthcare data in the cloud using ciphertext-policy attribute-based encryption, *2012 International Conference on Data Engineering Workshops (ICDEW)*, Arlington, Virginia, USA, 2012, pp. 143–146. https://doi.org/10.1109/ICDEW.2012.68

[6] C. Guo, R. Zhuang, Y. Jie, Y. Ren, T. Wu, K.-K. R. Choo, Fine-grained database field search using attribute-based encryption for e-healthcare clouds, *Journal of Medical Systems*, Vol. 40, No. 11, pp. 1–8, November, 2016. https://doi.org/10.1007/s10916-016-0588-0

[7] R. S. Sharon, R. J. Manoj, E-health care data sharing into the cloud based on deduplication and file hierarchical encryption, *2017 International Conference on Information Communication and Embedded Systems (ICICES)*, Chennai, Tamil Nadu, India, 2017, pp. 1–6. https://doi.org/10.1109/ICICES.2017.8070739

[8] K. Sowjanya, M. Dasgupta, A ciphertext-policy attribute based encryption scheme for wireless body area networks based on ECC, *Journal of Information Security and Applications*, Vol. 54, Article No. 102559, October, 2020. https://doi.org/10.1016/j.jisa.2020.102559

[9] Y. Tian, Y. Peng, X. Peng, H. Li, An attribute-based encryption scheme with revocation for fine-grained access control in wireless body area networks, *International Journal of Distributed Sensor Networks*, Vol. 10, No. 11, Article No. 259798, November, 2014. https://doi.org/10.1155/2014/259798

[10] A. Z. Alshamsi, E. S. Barka, Implementation of energy efficient/lightweight encryption algorithm for wireless body area networks, *2017 International Conference on Informatics, Health and Technology (ICIHT)*, Riyadh, Saudi Arabia, 2017, pp. 1–7. https://doi.org/10.1109/ICIHT.2017.7899139

[11] X. Wang, J. Zhang, E. M. Schooler, M. Ion, Performance evaluation of attribute-based encryption: Toward data privacy in the IoT, *2014 International Conference on Communications (ICC)*, Sydney, NSW, Australia, 2014, pp. 725–730. https://doi.org/10.1109/ICC.2014.6883405

[12] M. Rasori, P. Perazzo, G. Dini, A lightweight and scalable attribute-based encryption system for smart cities, *Computer Communications*, Vol. 149, pp. 78–89, January, 2020. https://doi.org/10.1016/j.comcom.2019.10.005

[13] M. Rasori, P. Perazzo, G. Dini, ABE-cities: An attribute-based encryption system for smart cities, *2018 International Conference on Smart Computing (SMARTCOMP)*, Taormina, Sicily, Italy, 2018, pp. 65–72. https://doi.org/10.1109/SMARTCOMP.2018.00075

[14] N. Chen, M. Gerla, D. Huang, X. Hong, Secure, selective group broadcast in vehicular networks using dynamic attribute based encryption, *2010 IFIP Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, Juan Les Pins, France, 2010, pp. 1–8. https://doi.org/10.1109/MEDHOCNET.2010.5546877

[15] D. Huang, M. Verma, ASPE: Attribute-based secure policy enforcement in vehicular ad hoc networks, *Ad Hoc Networks*, Vol. 7, No. 8, pp. 1526–1535, November, 2009. https://doi.org/10.1016/j.adhoc.2009.04.011

[16] K. N. Qureshi, F. Bashir, S. Iqbal, Cloud computing model for vehicular ad hoc networks, *2018 International Conference on Cloud Networking (CloudNet)*, Tokyo, Japan, 2018, pp. 1–3. https://doi.org/10.1109/CloudNet.2018.8549536

[17] K. Ren, C. Wang, Q. Wang, Security challenges for the public cloud, *IEEE Internet Computing*, Vol. 16, No. 1, pp. 69–73, January-February, 2012. https://doi.org/10.1109/MIC.2012.14

[18] B. Waters, Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization, *International Workshop on Public Key Cryptography*, Taormina, Italy, 2011, pp. 53–70. https://doi.org/10.1007/978-3-642-19379-8_4

[19] M. Chase, S. S. Chow, Improving privacy and security in multi-authority attribute-based encryption, *Proceedings of the 16th ACM Conference on Computer and Communications Security*, Chicago, Illinois, USA, 2009, pp. 121–130. https://doi.org/10.1145/1653662.1653678

[20] S. S. Chow, Removing escrow from identity-based encryption, *International Workshop on Public Key Cryptography*, Irvine, CA, USA, 2009, pp. 256–276. https://doi.org/10.1007/978-3-642-00468-1_15

[21] M. Belenkiy, J. Camenisch, M. Chase, M. Kohlweiss, A. Lysyanskaya, H. Shacham, Randomizable proofs and

delegatable anonymous credentials, *Annual International Cryptology Conference (CRYPTO)*, Santa Barbara, CA, USA, 2009, pp. 108–125. https://doi.org/10.1007/978-3-642-03356-8_7

[22] S. Wang, K. Liang, J. K. Liu, J. Chen, J. Yu, W. Xie, Attribute-based data sharing scheme revisited in cloud computing, *IEEE Transactions on Information Forensics and Security*, Vol. 11, No. 8, pp. 1661–1673, August, 2016. https://doi.org/10.1109/TIFS.2016.2549004

[23] U. S. Varri, S. K. Pasupuleti, K. Kadambari, Key-escrow free attribute-based multi-keyword search with dynamic policy update in cloud computing, *2020 International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*, Melbourne, VIC, Australia, 2020, pp. 450–458. https://doi.org/10.1109/CCGrid49817.2020.00-48

[24] J. Hur, Improving security and efficiency in attribute-based data sharing, *IEEE Transactions on Knowledge and Data Engineering*, Vol. 25, No. 10, pp. 2271–2282, October, 2013. https://doi.org/10.1109/TKDE.2011.78

[25] A. Shamir, How to share a secret, *Communications of the ACM*, Vol. 22, No. 11, pp. 612–613, November, 1979. https://doi.org/10.1145/359168.359176

[26] J.-P. Berrut, L. N. Trefethen, Barycentric lagrange interpolation, *SIAM Review*, Vol. 46, No. 3, pp. 501–517, September, 2004. https://doi.org/10.1137/S0036144502417715

[27] A. Beimel, *Secure Schemes for Secret Sharing and Key Distribution*, Technion - Israel Institute of Technology, 1996.

[28] J. Bethencourt, A. Sahai, B. Waters, Ciphertext-policy attribute-based encryption, *2007 IEEE Symposium on Security and Privacy (SP)*, Berkeley, CA, USA, 2007, pp. 321–334. https://doi.org/10.1109/SP.2007.11

[29] S. Goldwasser, S. Micali, Probabilistic encryption, *Journal of Computer and System Sciences*, Vol. 28, No. 2, pp. 270–299, April, 1984. https://doi.org/10.1016/0022-0000(84)90070-9

[30] N. Attrapadung, B. Libert, E. de Panafieu, Expressive key-policy attribute-based encryption with constant-size ciphertexts, *International Workshop on Public Key Cryptography*, Taormina, Italy, 2011, pp. 90–108.

[31] Z. Li, J. Higgins, M. Clement, Performance of finite field arithmetic in an elliptic curve cryptosystem, *2001 International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, Cincinnati, OH, USA, 2001, pp. 249–256. https://doi.org/10.1109/MASCOT.2001.948875

[32] K. Takashima, Scaling security of elliptic curves with fast pairing using efficient endomorphisms, *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. 90, No. 1, pp. 152–159, January, 2007. https://doi.org/10.1093/ietfec/e90-a.1.152

[33] K. Lauter, The advantages of elliptic curve cryptography for wireless security, *IEEE Wireless Communications*, Vol. 11, No. 1, pp. 62–67, February, 2004. https://doi.org/10.1109/MWC.2004.1269719

[34] C.-I. Fan, Y.-F. Tseng, C.-C. Feng, CCA-secure attribute-based encryption supporting dynamic membership in the standard model, *2021 IEEE Conference on Dependable and Secure Computing (DSC)*, Aizuwakamatsu, Fukushima, Japan, 2021, pp. 1–8. https://doi.org/10.1109/DSC49826.2021.9346247

Biographies

Jheng-Jia Huang was born in Kaohsiung, Taiwan. In 2020, he joined the Department of Information Management at National Taiwan University of Science and Technology, Taipei, Taiwan. His current research interests include cloud computing security, social network security and authentication, network and communication security, information security, and applied cryptography.

Guan-Yu Chen was born in Kaohsiung, Taiwan. He received the B.S. degree in Department of Statistics from the National Chengchi University, Taipei, Taiwan, in 2018. He is currently working toward the Ph.D. degree in Department of Information Management, National Taiwan University of Science and Technology, Taipei, Taiwan.

Shao-Wei Tung was born in Keelung, Taiwan. He received the M.S. degree in Department of Information Management, National Taiwan University of Science and Technology, Taipei, Taiwan, in 2022. He is currently working for the Taiwan Semiconductor Manufacturing Company.