JOURNAL ON COMMUNICATIONS

ISSN:1000-436X

Scopus®

REGISTERED

www.jocs.review

"Attribute-Based Encryption and Access Control in Cloud Computing: A Survey and a Temporal ABE Framework"

Dr. Vinay Warad[1], Shireen Tabbassum[2], Dr. Ruksar Fatima[3]

[1] **Assistant Professor, Khaja Bandanawaz University**
[2] **Assistant Professor, Khaja Bandanawaz University**
[3] **Professor, Khaja Bandanawaz University**

Corresponding Author: Dr. Ruksar Fatima[3]

Abstract

Background:
Cloud computing has become a vital infrastructure for modern data storage and sharing due to its scalability, cost-effectiveness, and accessibility. However, the untrusted nature of cloud environments raises significant concerns regarding data confidentiality and fine-grained access control. Attribute-Based Encryption (ABE) has emerged as a promising cryptographic technique, enabling secure data sharing by embedding access policies directly into ciphertext. While numerous ABE schemes exist, challenges remain in scalability, dynamic policy updates, and resistance to key compromise.

Results:
This paper surveys traditional access control models—Discretionary Access Control (DAC), Mandatory Access Control (MAC), Role-Based Access Control (RBAC), and Attribute-Based Access Control (ABAC)—and evaluates their applicability in cloud environments. A detailed classification of ABE schemes, including CP-ABE, KP-ABE, Hierarchical ABE, Non-Monotonic ABE, and Outsourced/Distributed ABE, is presented with comparative analysis. Building on this, we propose an enhanced **Temporal Attribute-Based Encryption (T-ABE)** framework that integrates time-varying encryption with ABAC principles. The system decouples ciphertext from policies using an XML-based metadata file, ensuring dynamic and modular policy management. This design improves resilience against static key analysis, simplifies revocation, and enhances fine-grained access control in distributed cloud environments.

Conclusion:
The proposed T-ABE scheme addresses key limitations of conventional ABE models by introducing temporal constraints and metadata-driven access control. This makes it a more adaptive, secure, and scalable solution for real-world cloud applications. Future work will focus on implementing and benchmarking the scheme against existing ABE and ABAC models to evaluate its efficiency in policy updates, revocation handling, and access enforcement.

"The key contribution of this work lies in proposing a novel **Temporal Attribute-Based Encryption (T-ABE)** framework that integrates time-varying encryption with metadata-driven policy management, thereby enabling secure, flexible, and fine-grained access control in cloud computing environments."

Keywords

Cloud computing; Attribute-Based Encryption; Access control; Temporal ABE; Fine-grained security; Metadata; Policy management; Data confidentiality

Introduction

Cloud computing has become a widely adopted paradigm, offering usage-based services that provide flexible access to various network resources, including networked storage [24, 35]. In typical scenarios, data owners upload their data to the cloud, associating it with an access policy that governs its use. However, any user who satisfies this policy can potentially access the data, posing a significant challenge to privacy and security [23].

To address this, cryptographic techniques have been introduced, with encryption being a fundamental method for securing data in untrusted cloud environments [37]. Nevertheless, conventional encryption alone is insufficient—particularly in cases of key compromise, where attackers may still gain unauthorized access to encrypted data. Moreover, when the ciphertext must be shared among users and the cloud provider lacks decryption privileges, the complexity of secure data sharing increases [13].

An effective solution to these challenges is Attribute-Based Encryption (ABE). ABE introduces a paradigm shift by embedding access policies directly into the ciphertext, allowing for fine-grained access control based on user attributes rather than identity [1–3]. In ABE systems, access policies can enforce role-based, content-based, or attribute-based controls, thereby enabling self-enforcing data access through cryptographic means [9, 10, 22].

1.1 Background

This section outlines the mathematical concepts essential to Attribute-Based Encryption (ABE), focusing on bilinear maps and associated security assumptions that ensure cryptographic strength in cloud storage environments [5, 6].

BilinearMaps

The concept of Identity-Based Encryption (IBE) was introduced by Shamir in 1985 [37], but practical key generation remained a significant hurdle. Traditional approaches such as RSA and the Diffie–Hellman (DH) protocol failed to yield secure and efficient IBE systems. Bilinear maps, however, have proven instrumental in the development of pairing-based cryptographic systems, including IBE and ABE schemes [1, 2, 5].

Let G_1, G_2 and G_T be cyclic groups of prime order q,

$$e: G_1 \times G_2 \longrightarrow G_T$$

The bilinear map **e** satisfies the following properties:

Bilinearity: For all a, b ∈ Zq,

$$e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$$

Computability: The function e(u,v), is efficiently computable for all

$$u \in G_1, v \in G_2$$

Non-degeneracy: $e(g_1,g_2) \neq 1_{GT}$, i.e., the map does not send all input pairs to the identity element of G_T. Pairings that satisfy all three properties are referred to as **admissible bilinear maps** and serve as the foundation for numerous cryptographic protocols.

Security Assumptions
The security of bilinear-map-based encryption schemes relies on hard computational problems in cyclic groups. Two foundational assumptions are described below [5]:
Computational Diffie–Hellman (CDH) Assumption:
Let G be a cyclic group of prime order q with generator g. Given g, g^a and g^b for random a,b∈Zq , it is computationally infeasible to compute g^{ab}.

Bilinear Diffie–Hellman (BDH) Assumption:
Given $g, g^a, g^b, g^c \in G$ and a bilinear map $e: G_1 \times G_2 \longrightarrow G_T$ it is difficult to compute

$e(g,g)^{abc}$

Decisional Bilinear Diffie–Hellman (DBDH) Assumption:
Given the tuple (g, g^a, g^b, g^c, T), it is hard to distinguish whether for a randomly chosen z∈Zq.

Several well-known ABE systems, such as the Fuzzy IBE scheme [1] and Key-Policy ABE (KP-ABE) [2], are proven secure under the DBDH assumption. These mathematical underpinnings are critical for the construction and security of ABE schemes, particularly in cloud environments where data confidentiality and fine-grained access control are essential [24, 26].

The concept of Attribute-Based Encryption (ABE) was introduced by Amit Sahai and Brent Waters in 2005 [1]. In ABE, encryption and decryption are governed by attributes rather than explicit user identities. A set of descriptive attributes functions as an identity for generating secret keys and enforcing access policies [2, 3].

ABE systems are typically derived from the Fuzzy Identity-Based Encryption (Fuzzy-IBE) model [1] and consist of four core algorithms: Setup, Key Generation, Encryption, and Decryption. These modules are fundamental to almost all cryptographic schemes employing ABE and have evolved into more scalable and expressive variants such as CP-ABE [3], KP-ABE [2], RABE [19, 20], and HABE [9, 18].

Let the security parameter κ determine the bit-length of group elements in the bilinear pairing setup.

The algorithms are formally described as follows:

Algorithm 1: ABE Scheme

1. Setup: Set up (κ)→ (PK, MK)

Initializes the bilinear group environment using the security parameter κ, and returns the public key (PK) and master key (MK). The parameter K denotes the size of the attribute universe.

2. Key Generation: $\text{KeyGen}(MK, \tau) \rightarrow SK$

For a given access structure τ (e.g., a threshold tree), and the master key MK, this algorithm generates a secret key SK corresponding to a user's attribute set S.

3. Encryption: $\text{Encrypt}(M, S', PK) \rightarrow CT$

Encrypts the plaintext message M under a set of attributes S′, producing the ciphertext CT. Only users whose attribute set satisfies the embedded access policy can decrypt the message.

4. Decryption: $\text{Decrypt}(CT, SK) \rightarrow M \text{ or } \perp$

Uses the ciphertext CT and secret key SK to retrieve the message M. If the user's attributes do not satisfy the access structure, the decryption returns ⊥ (null or failure).

To strengthen system security, researchers have proposed extensions to the basic ABE scheme by integrating additional cryptographic components or complexity layers—enhancing fine-grained access control and computational efficiency.

1.3 Limitations of ABE

Despite its advantages, ABE systems face several limitations [15, 16, 19]:

1. **Computational Overhead:**
   Supporting multiple user categories with diverse attribute sets can significantly increase computational cost during encryption and decryption [10, 21].
2. **Limited Expressiveness:**
   Basic ABE lacks fine-grained control, such as the ability to express threshold-based conditions or numerical range queries natively [17, 25].

1.4 Classification of ABE

Attribute-Based Encryption (ABE) schemes are broadly classified into two main categories: Key-Policy ABE (KP-ABE) and Ciphertext-Policy ABE (CP-ABE), as illustrated in Fig. 1 [2, 3, 4].

1) Key-Policy ABE (KP-ABE)
The KP-ABE scheme was introduced by Vipul Goyal, Omkant Pandey et al. [2], and is based on the Decisional Bilinear Diffie–Hellman (DBDH) assumption [5]. It supports fine-grained access control by embedding the access structure in the user's decryption key, while the ciphertext is labelled with attributes.

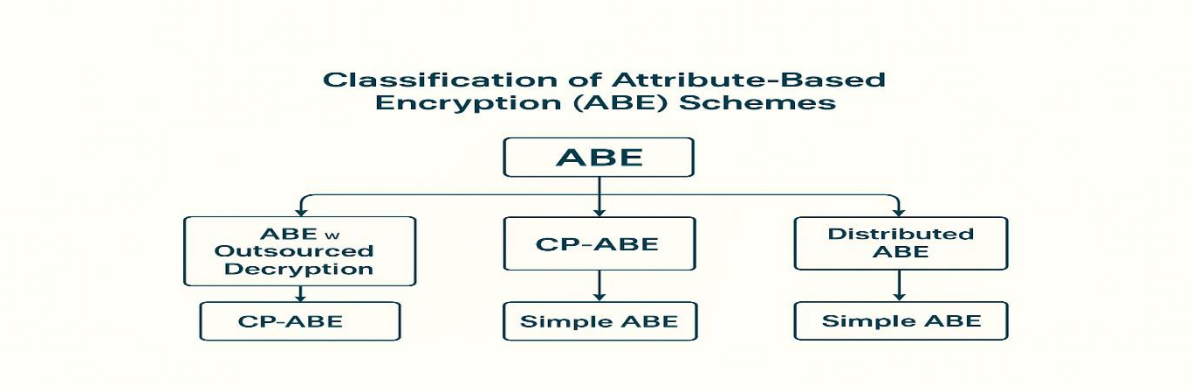


Fig. 1. Classification of Attribute-Based Encryption (ABE) Schemes

A KP-ABE system consists of the following algorithms:

Algorithm 2: Key-Policy ABE

Setup: $\text{Setup}(k) \rightarrow (PK, MK)$
Encryption: $\text{Encrypt}(M, S, PK) \rightarrow E$
Key Generation: $\text{KeyGen}(A, MK) \rightarrow D$
Decryption: $\text{Decrypt}(E, S, D) \rightarrow M \text{ if } S \in A; \text{ else } \perp$

Where:
K: Security parameter
M: Message
S: Set of attributes
A: Access structure
PK: Public key
MK: Master key
E: Ciphertext
D: Decryption key

In KP-ABE, a user's secret key is associated with an access structure, and decryption is possible only if the attribute set SSS of the ciphertext satisfies the access structure AAA embedded in the decryption key. However, the **limitation** of KP-ABE lies in the lack of control by the data owner, who cannot specify which users are allowed to decrypt the data.

2) Ciphertext-Policy ABE (CP-ABE)
CP-ABE was proposed by Bethencourt, Sahai, and Waters in 2007 [3]. Unlike KP-ABE, in CP-ABE the access structure is embedded in the ciphertext, and a user's secret key is associated with a set of attributes. This model allows the data owner (encryptor) to define an access policy that explicitly determines who can decrypt the data [4].

CP-ABE involves the following algorithms:

Algorithm 3: Ciphertext-Policy ABE

Setup: $\text{Setup}(k) \rightarrow (PK, MK)$
Encryption: $\text{Encrypt}(M, S, A) \rightarrow E$
Key Generation: $\text{KeyGen}(S, MK) \rightarrow SK$
Decryption: $\text{Decrypt}(PK, E, SK) \rightarrow M$
Delegation (Optional): $\text{Delegate}(SK, s) \rightarrow SKS$

Where:
K: security parameter
M: message (plaintext)
A: access structure
S: attribute set
$S \subseteq S$: delegated attribute subset
PK: public key
MK: master key
SK: secret key
E: ciphertext

The **Delegate** algorithm allows a user to generate a restricted secret key SKS from their original key SK, such that it can only be used for a subset of attributes {S}. This provides flexible key management and further supports fine-grained access control.

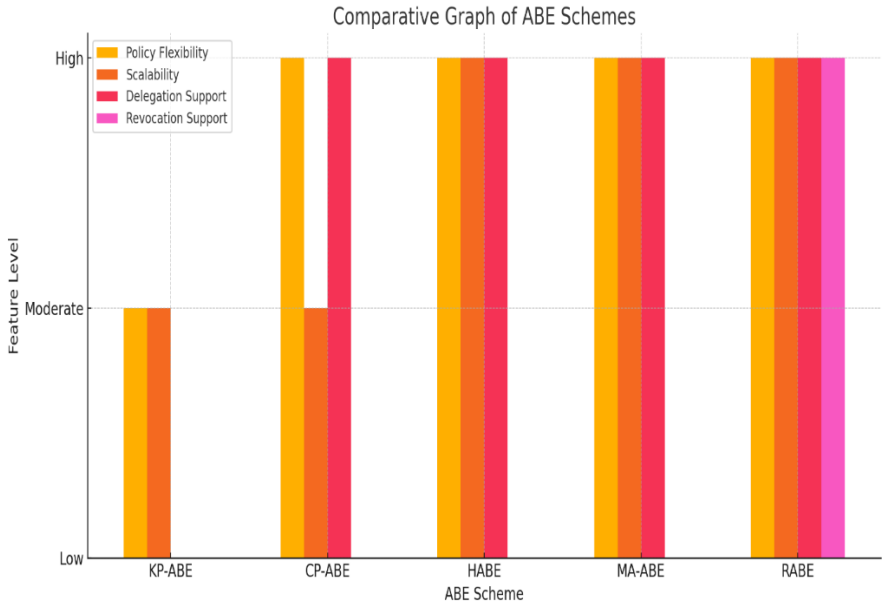
Unlike KP-ABE, CP-ABE gives the data owner **explicit control** over who can decrypt the ciphertext, making it more suitable for real-world cloud applications requiring policy-enforced data sharing.

Table I: Comparison Between KP-ABE and CP-ABE Schemes

Feature	Key-Policy ABE (KP-ABE)	Ciphertext-Policy ABE (CP-ABE)
Introduced by	Goyal et al., 2006	Bethencourt et al., 2007
Access Structure Location	Embedded in the decryption key	Embedded in the ciphertext
Attribute Association	Ciphertext is labeled with a set of attributes	User's secret key is associated with a set of attributes
Control over Access	Data owner has limited control over who can decrypt	Data owner specifies explicit access policy
Flexibility	Less flexible; access structure defined at key generation	More flexible; access policy defined at encryption
Suitability	Better for system-defined roles or centralized access control	Better for user-defined roles and decentralized environments
Support for Delegation	Limited or not natively supported	Supports key delegation (optional algorithm)
Security Assumption	Based on Decisional Bilinear Diffie–Hellman (DBDH)	Also based on DBDH assumption
Application Example	Institution-based access policies	Secure document sharing, EHR systems

Table II: Comparative Analysis of Advanced ABE Variants

Feature / Scheme	KP-ABE	CP-ABE	HABE	MA-ABE	RABE
Access Structure	In decryption key	In ciphertext	Hierarchical	Distributed (multi-authority)	In ciphertext/key
Policy Flexibility	Moderate	High	Moderate to High	High	High
Scalability	Moderate	Moderate	High	High	High
Delegation Support	Limited	Yes (optional)	Yes	Yes	Yes
User Revocation	Not Supported	Not Native	Not Native	Not Native	**Yes (core feature)**
Authority Structure	Single authority	Single authority	Hierarchical authorities	Multiple independent authorities	Single or federated
Use Case Example	Education system	Health data sharing	Military / Corporate orgs	E-governance, Research DB	Dynamic cloud environments



Here is a comparative Bar chart illustrating the key features across five ABE schemes (KP-ABE, CP-ABE, HABE, MA-ABE, and RABE). Each feature—Policy Flexibility, Scalability, Delegation Support, and Revocation Support—is evaluated for its implementation level: Low, Moderate, or High. Let me know if you'd like this exported as an image or inserted into a report format.

1.5. ABE with Non-Monotonic Access

Ostrovsky et al. proposed an advanced Attribute-Based Encryption (ABE) scheme in 2007 that supports non-monotonic access structures [4]. Traditional ABE schemes primarily allow monotonic access policies (i.e., policies based only on positive attributes). However, in many real-world scenarios, it is necessary to express negative constraints (e.g., "access if the user does not have attribute X"). This non-monotonic ABE (NM-ABE) scheme addresses this

limitation and has been referenced as a precursor to more expressive policy-aware cryptographic models [15, 17, 25].

KeyInnovation
The scheme allows for the specification of negative attributes within the access structure, thereby supporting more expressive and fine-grained access control mechanisms in the cloud environment [4].

Algorithm 3: Non-Monotonic ABE

Step	Function	Description
1.	Setup(d)	Initializes system parameters for d attributes. Returns public key PK and master key MK.
2.	Encrypt(M, γ, PK)	Encrypts a message M ∈ GT under a set of d attributes γ ⊂ Z*_p. Randomly selects s ∈ Z_p and generates ciphertext CT.
3.	KeyGen(Â, MK, PK)	Generates a private key D for access structure Â. The key enables decryption only if ciphertext attributes satisfy Â.
4.	Decrypt(CT, D)	Uses the private key D to decrypt CT. Returns plaintext M if access policy is met; otherwise, returns NULL.

Where

d – Number of attributes in the system.
γ – Attribute set associated with ciphertext.
PK – Public key.
MK – Master key.
Â – Access structure (including negative attributes).
CT – Ciphertext.
D – Private key.

Advantages:

- Supports expressive access control with **negative constraints**.
- More realistic for real-world cloud-based access requirements.

Limitations:

- Slightly **increased computational complexity** due to the evaluation of non-monotonic access structures.
- Implementation and key management can be more challenging than in monotonic-only schemes.

1.6. Hierarchical Attribute-Based Encryption (HABE)

Hierarchical Attribute-Based Encryption (HABE) is a hybrid encryption model that combines the benefits of Hierarchical Identity-Based Encryption (HIBE) and Ciphertext-Policy Attribute-Based Encryption (CP-ABE). This model was proposed by Wan, Liu, and Deng to

enhance scalability and efficiency in large organizations with multiple administrative domains [9]. Recent advancements have extended this architecture to support distributed trust and cross-domain delegation [18, 19, 27].

The HABE architecture introduces the following hierarchical structure [9]:

- **Root Master (RM)** – Central authority that initializes the system.
- **Third Trusted Party (TTP)** – Responsible for domain management.
- **Domain Masters (DMs)** – Hierarchically organized authorities that manage users and issue keys.
- **Users** – Assigned attributes by DMs and granted access based on policies.

Algorithm 4: HABE Scheme

Step	Function	Description
1.	$\texttt{Setup(K)} \rightarrow \texttt{(params, MK}_0\texttt{)}$	RM generates system parameters `params` and root master key $\texttt{MK}_0$.
2.	$\texttt{CreateDM(params, MK}_i\texttt{, PK}_{i+1}\texttt{)} \rightarrow \texttt{MK}_{i+1}$	Creates a new domain master at the next level with updated master key.
3.	$\texttt{CreateUser(params, MK}_i\texttt{, PK\_u, PK\_a)} \rightarrow \texttt{(SK}_{i,u}\texttt{, SK}_{i,u,a}\texttt{)}$	Issues secret keys for a user and their attributes.
4.	`Encrypt(params, f, A, {PK$_a$	a ∈ A})→CT`
5.	`Decrypt(params, CT, SK$_{i,u}$, {SK$_{i,u,a}$	a ∈ CC$_j$})→f`

Where

$\texttt{K}$ – Security parameter
`params` – Public system parameters
$\texttt{MK}_0$ – Root master key
`PK`, `SK` – Public and secret keys
`f` – Encrypted file or message
`A` – Access structure (DNF form)
$\texttt{CC}_j$ – *j*-th conjunctive clause in the policy

Setup Details:

The **Root Master (RM)**:

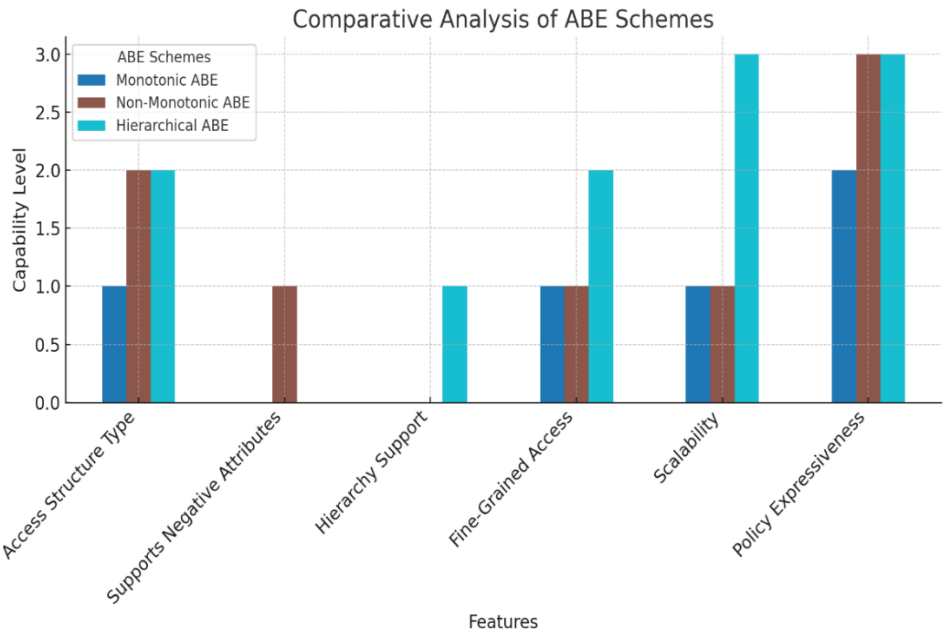
- Selects master key $\texttt{mk}_0 \in \texttt{Z\_q}$.
- Chooses bilinear groups $\texttt{G}_1, \texttt{G}_2$ of order $\texttt{q}$ with a bilinear map $\texttt{e: G}_1 \times \texttt{G}_1 \rightarrow \texttt{G}_2$.
- Defines two hash functions:
  $\texttt{H}_1\texttt{: \{0,1\}*} \rightarrow \texttt{G}_1$ and $\texttt{H}_2\texttt{: G}_2 \rightarrow \texttt{\{0,1\}}^n$
- Chooses a generator $\texttt{P}_0 \in \texttt{G}_1$ and computes public parameters.

Advantages:

- Supports hierarchical user management.
- Fine-grained and scalable access control.
- Enables decentralized administration across multiple domains.

Limitations:

- Increased system complexity due to multi-level key delegation.
- Higher overhead in key management and encryption compared to flat ABE models.



Graph1: Comparing Monotonic ABE, Non-Monotonic ABE, and **Hierarchical ABE across key features such as:**

- **Access Structure Type**
- **Support for Negative Attributes**
- **Hierarchy Support**
- **Fine-Grained Access**
- **Scalability**
- **Policy Expressiveness**

Each bar represents the capability level of each scheme for a specific feature (0 = Not Supported, 1 = Basic Support, 2 = Good, 3 = Strong). Let me know if you'd like this in tabular form for your IEEE paper or need to include more schemes like KP-ABE or CP-ABE.

2. Methods

2.1. Access Control Models

Access control plays a pivotal role in ensuring data security within cloud storage systems. It refers to the mechanisms and policies that regulate which users (subjects) are permitted to

access which resources (objects), under what conditions, and to what extent. Additionally, access control mechanisms may monitor and log attempts to access the system, thereby enhancing auditability and accountability [8].

Historically, access control models have evolved in response to differing requirements in military versus commercial domains. This evolution has led to the establishment of several foundational access control paradigms:

- **Mandatory Access Control (MAC):** Commonly used in high-security environments such as the military, MAC enforces access based on predefined security classifications and labels. Users cannot alter access permissions, which are centrally controlled [7].
- **Discretionary Access Control (DAC):** More flexible than MAC, DAC allows resource owners to control access to their resources. It is suitable for collaborative environments but can be vulnerable to privilege escalation [7, 13].
- **Role-Based Access Control (RBAC):** RBAC introduces the concept of roles, which are assigned specific permissions. Users acquire access rights based on their roles rather than their individual identity, improving scalability and manageability [13].

These traditional models are often classified as identity-based access control models, where both users and resources are identified using unique identifiers, and access decisions are made accordingly. While effective in static or well-defined distributed environments, these models show limitations in dynamic and scalable cloud ecosystems, where users and services are frequently added or modified [8].

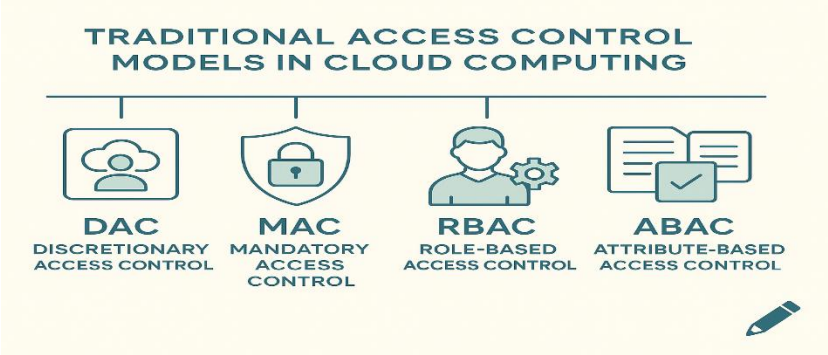


Fig. 3. Traditional Access Control Models in Cloud Computing

Each model presents unique advantages depending on the context and security requirements. However, the dynamic nature of cloud computing demands more fine-grained, flexible, and scalable access control mechanisms—paving the way for attribute-based and policy-based models.

2.2. Discretionary Access Control (DAC) Model

The Discretionary Access Control (DAC) model grants users the ability to control access to their owned resources or information, typically based on user identity or group membership [7, 13]. Under this model, the owner of an object (such as a file or directory) has the discretion to determine which users or groups are granted access and what operations they may perform [8].

DAC is inherently more flexible but generally considered less secure than other models, especially when compared to Mandatory Access Control (MAC) [7]. This is because users with ownership rights can propagate access to others, potentially resulting in unintentional privilege escalation or data leakage. As a result, DAC is most suitable for environments where strict security policies are not mandatory [13].

Despite its limitations, DAC remains widely adopted in commercial operating systems, including UNIX and Microsoft Windows platforms, due to its ease of implementation and user-centric design [8].

There are primarily two methods for implementing DAC [13]:

- **Identity-Based Access Control:** Access is granted directly to user identities or groups.
- **Access Control Matrix or List (ACL):** Defines permissions for each subject-object pair in matrix form, with entries specifying allowed operations.

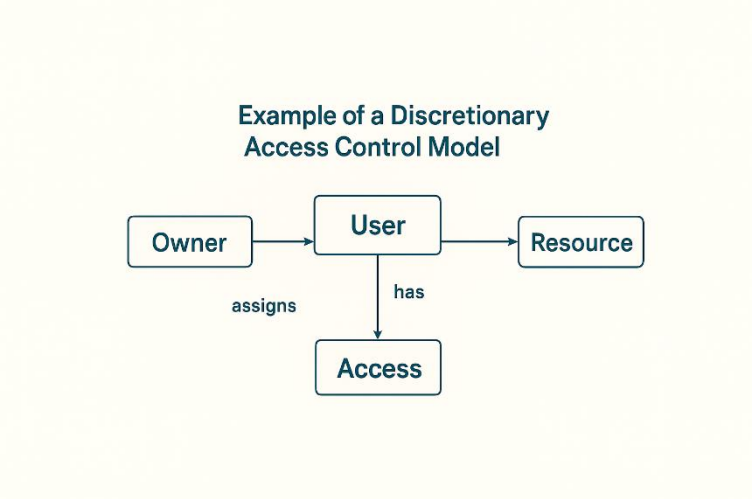


Fig. 4. Example of a Discretionary Access Control Model

The DAC model provides essential access control capabilities in systems with **moderate security requirements** and supports a **decentralized access policy**, where users play an active role in permission management.

2.3. Mandatory Access Control (MAC) Model

The Mandatory Access Control (MAC) model enforces access policies determined by a central authority, rather than leaving control to individual users [7, 8]. Unlike DAC, where users can assign permissions, MAC restricts access based on predefined rules associated with security classifications of both subjects (users/processes) and objects (files/data) [7].

In MAC, every subject and object is assigned a security label or classification (also called an access class), which defines its sensitivity level. Access decisions are based on these classifications and the dominance relationship between them. For example, if a resource RS is classified as “Top Secret,” only users with the appropriate clearance level—equal to or higher

than “Top Secret”—are permitted to access it. Users with a lower classification will be denied access, regardless of ownership or group membership [7].

The most foundational formalization of MAC was introduced by **Bell and LaPadula (1973)**, and it remains widely cited in the context of multilevel security systems [7]. The Bell–LaPadula Model enforces two core properties:

- **No Read Up (Simple Security Property):** A subject cannot read data at a higher security level than its own.
- **No Write Down (*-Property):** A subject cannot write data to a lower security level, preventing leakage of sensitive information.

These principles ensure strict control over information flow, maintaining confidentiality by preventing unauthorized data access or leakage across security boundaries [7, 8].

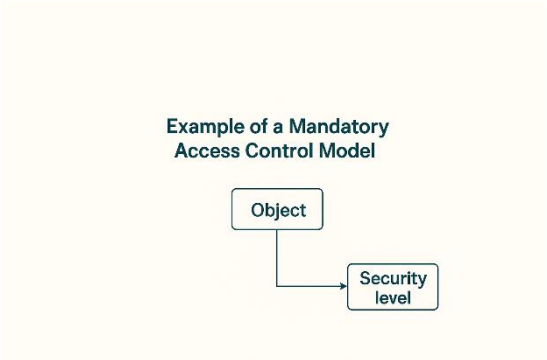


Fig. 5. Example of a Mandatory Access Control Model

The MAC model is best suited for **military, government, and enterprise environments** that require rigid access policies and strong confidentiality guarantees. Its enforcement of centralized control and classification-based access makes it more secure—but also less flexible—than DAC.

2.4. Role-Based Access Control (RBAC)

Role-Based Access Control (RBAC)

Role-Based Access Control (RBAC) is a widely adopted model in cloud environments where access decisions are based on a user’s role within an organization, rather than the user's individual identity [13]. The underlying principle of RBAC is that "a subject’s responsibility is more important than who the subject is" [8, 13].

In this model, roles are assigned to users, and permissions are associated with roles, not directly with users [13]. A user can inherit multiple roles, and each role encapsulates a set of access rights needed to perform specific tasks. For instance, an employee may be assigned both the “Secretary” and “Employee” roles, each granting different levels of access to resources [8].

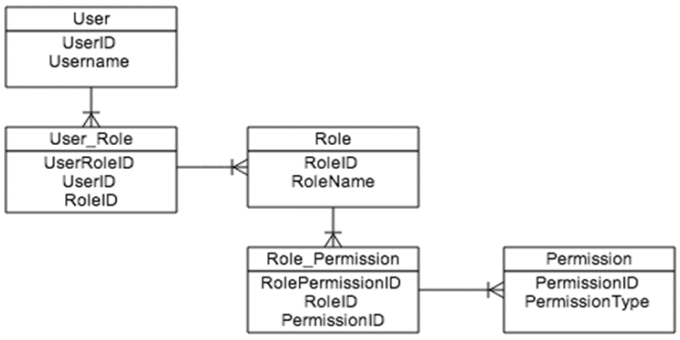


Fig. 6. A Basic Model of Role-Based Access Control

RBAC significantly simplifies **access control administration** in dynamic cloud environments by centralizing permission management. It ensures **least privilege**, meaning users can only access the information necessary for their role.

An evolution of RBAC is the **Task-Role Based Access Control (T-RBAC)** model, which introduces more granularity by assigning permissions to specific tasks rather than to roles. This approach enhances security and flexibility by linking access rights directly to workflow actions.

RBAC is particularly effective in **enterprise cloud systems** where user roles are well-defined and access control must adapt to organizational structures and job responsibilities. It also supports role hierarchies and constraints, allowing scalable and policy-compliant access management.

2.5. Attribute-Based Access Control (ABAC)

Attribute-Based Access Control (ABAC) is a policy-driven access control model where access decisions are made based on the attributes of the subject (user), resource (object), action, and environment [8]. This fine-grained control mechanism dynamically evaluates Boolean policy rules over a wide range of attributes to determine access rights [22].

Unlike traditional models such as RBAC or ACLs, where access permissions are tied directly to roles or identities, ABAC separates the policy from the user and object, allowing for greater flexibility and scalability [8, 23]. Policies in ABAC are expressed as complex Boolean expressions that can evaluate combinations of user attributes (e.g., department, clearance level), resource attributes (e.g., sensitivity, owner), and environmental attributes (e.g., time, location) [22].

While RBAC can be extended to achieve some objectives of ABAC, it often becomes inflexible when access control requirements evolve. For instance, changes in policies require identifying and modifying multiple roles or ACL entries, which is both error-prone and inefficient [8].

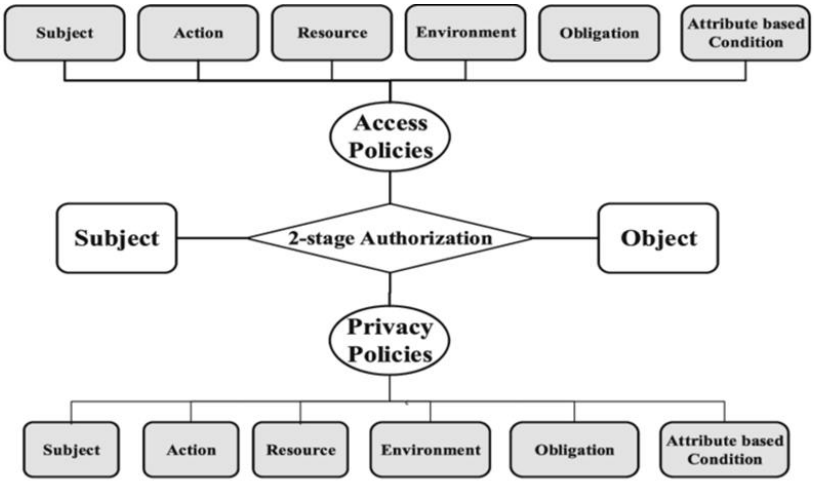
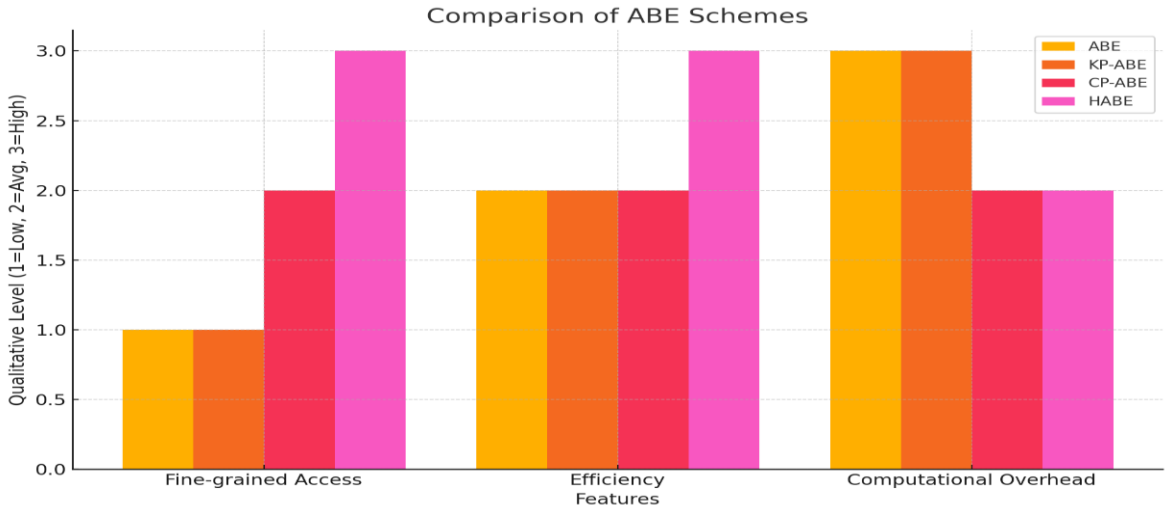


Fig. 7. A Conceptual Model of Attribute-Based Access Control

The **National Institute of Standards and Technology (NIST)** addressed this issue by releasing Special Publication (SP) 800-162**,** titled *"Guide to Attribute-Based Access Control (ABAC): Definition and Considerations"*. This document outlines the key functional components of ABAC and provides guidance for its deployment in large-scale enterprise environments. Importantly, ABAC facilitates secure data sharing while maintaining robust control over access decisions, without delving into the specifics of authentication mechanisms.

ABAC is particularly advantageous in cloud environments due to its **dynamic, context-aware** nature, enabling secure collaboration across heterogeneous domains and user populations.



Graph 2: Comparing ABE schemes based on qualitative features like fine-grained access, efficiency, and computational overhead

3. Results

To ensure robust and adaptive security for cloud-stored documents, we propose a modified Attribute-Based Encryption (ABE) scheme that incorporates a time-varying encryption mechanism [12, 19, 31]. This enhancement introduces temporal dynamics into the encryption process, making it more resistant to static key analysis and unauthorized access. The proposed

system will function under the Attribute-Based Access Control (ABAC) model, ensuring flexible and fine-grained access control [8, 22, 23].

Access permissions will be maintained separately in a metadata file, structured in XML format. This metadata file will be tightly linked with the encrypted document and processed in parallel, providing dynamic and controlled access management. By decoupling the encryption from access policies and storing metadata separately, the proposed framework offers enhanced security, modular policy updates, and easier revocation handling—particularly useful in untrusted or distributed cloud environments [19, 20, 27].

3.1. Introduction to the Proposed Work

The proposed scheme enhances cloud document security by modifying the standard Attribute-Based Encryption (ABE) method with time-varying encryption [12, 31]. Traditional ABE ties access to user attributes, but lacks resilience against temporal threats and static key exposure [19]. By integrating temporal dynamics (e.g., encryption keys that change over time), and separating access control logic into an XML-based metadata file, we enable fine-grained, dynamic access control suitable for distributed or untrusted cloud settings [8, 22, 23].

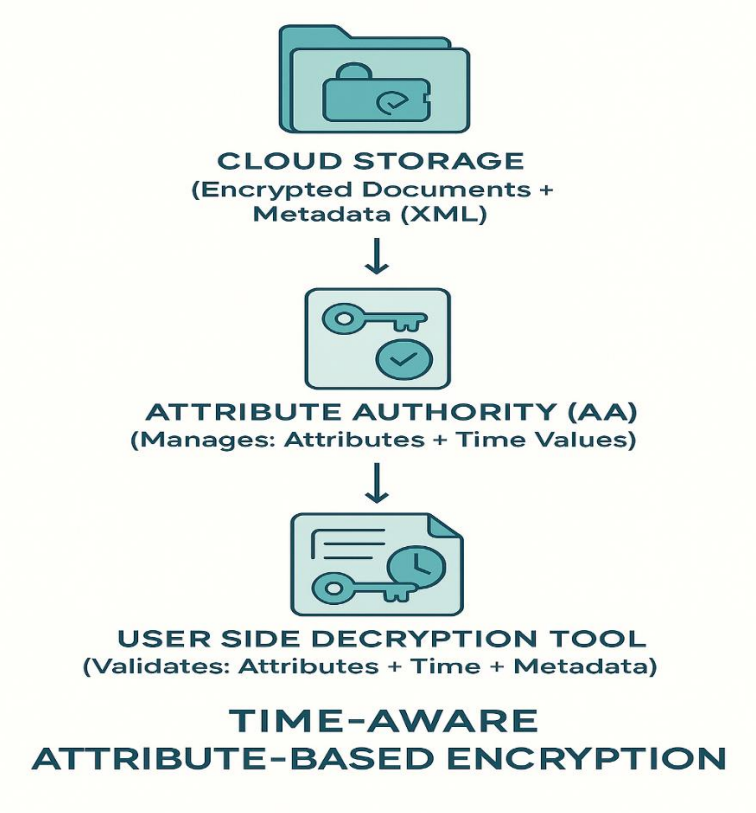


Fig. 8. A Conceptual Model to explain the Proposed Work

4. Discussion

4.1. Attribute-Based Cryptographic Access Control in the Cloud: University Scenario Explained

Attribute-Based Encryption (ABE) is a modern cryptographic approach that allows fine-grained access control to encrypted data stored in the cloud [1–3]. Instead of encrypting data for specific users, you encrypt data so that only users whose attributes match a policy can decrypt it [8, 22, 23].

Scenario:
A university uses cloud storage to host sensitive documents—like research papers, student records, and faculty contracts [9, 10].

Scenario Overview

- **Environment:** University uses cloud storage for sensitive documents.
- **Asset:** Documents like research papers, student records, and faculty contracts.
- **Objective:** Ensure only authorized users—based on their attributes—can decrypt and view documents [21, 23].

Attributes for Access Control:

- Role = {Student, Faculty, Admin}
- Department = {CSE, ECE, MECH}
- Access_Level = {Read, Write}
- Valid_Time = {Until_Date}

Example Use Case:
Dr. Ruksar (Role = Faculty, Department = CSE) needs access to a research document from Jan 1 to Dec 31, 2025. The document will be encrypted using ABE with those attributes. If Dr. Ruksar tries to access it after Dec 31, the time-varying key won't match, and decryption will fail—even if other attributes still hold true [12, 19, 27, 31].

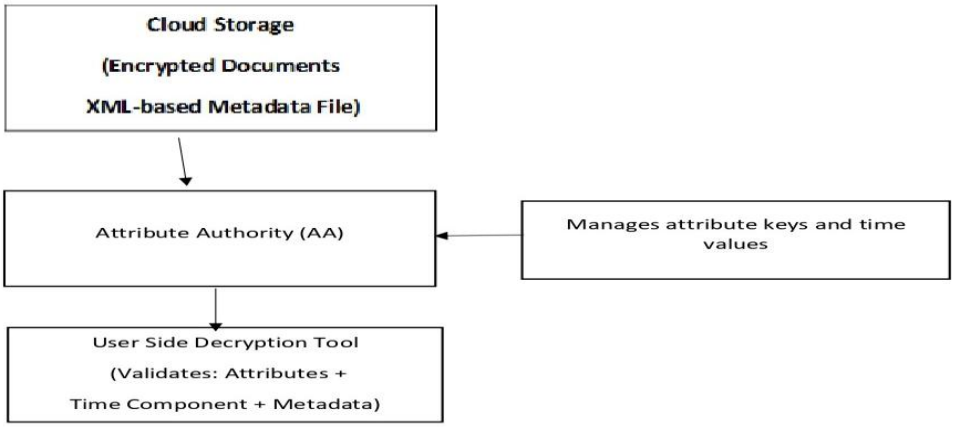
Visual Flow

- Admin defines access policy → "Faculty in CSE, time before Dec 31, 2025".
- Document is encrypted with ABE using that policy [3, 9].
- User (Dr. Ruksar) tries to decrypt:
    - If attributes match (Role = Faculty, Dept = CSE, Valid_Time = within range) → Access granted.
    - If Valid_Time expired, or any attribute doesn’t match → Access denied [12, 31].

4.2. Architecture Overview

- **ABE Encryption** applies at file level [1, 2, 10].
- **Metadata File (XML)** stores:

- List of permitted attributes
- Time range validity
- Revocation info [19, 20, 27].



How ABE Works in This Scenario

1. Defining Access Policies

- Every document is encrypted with an **access policy**.
  For example, a research paper may be encrypted so that only:

  "Role = Faculty AND Department = CSE AND Valid_Time ≤ Dec 31, 2025" can decrypt it.

- Policies can be as granular as needed (e.g., only ECE Students with Write permission).

2. User Attribute Keys

- Each user is issued cryptographic keys tied to their attributes (Role, Department, etc.).
- Dr. Ruksar would have keys indicating:
  - Role: Faculty
  - Department: CSE
  - Valid_Time: Jan 1 – Dec 31, 2025
  - (possibly more—such as Access_Level)

3. Encryption and Storage

- **Encryption:** When a document is uploaded, it's encrypted using the access policy. Anyone can download the ciphertext, but only holders of matching attribute keys can decrypt.
- **Cloud Storage:** The encrypted document is uploaded to the cloud, accessible to anyone—but encrypted.

4. Access Attempt and Decryption

- When Dr. Ruksar wants to access the file:
  - Her decryption key is checked against the document's policy.
  - If her attributes match (e.g., she is faculty, in CSE, and within the valid time), decryption succeeds, and she can read the document.
- **Example:**
  - If she tries to access the document on Dec 15, 2025:
    - Her Valid_Time attribute (up to Dec 31, 2025) is valid—access is granted.
  - If she tries to access the document on Jan 1, 2026:
    - Her Valid_Time has expired—decryption fails, even if Role and Department are correct.

Why This Approach Is Powerful

- **Fine-Grained Control:** You can enforce highly specific policies—down to roles, departments, and time windows.
- **Automatic Expiry:** Access can "turn off" automatically after expiration dates, even if the user's role or department hasn't changed.
- **No Cloud Trust Needed:** The cloud provider never sees or manages keys—only ciphertext. Sensitive data stays protected.

Key Points

- Users never get direct document access—only if their attributes match all required criteria.
- Time-varying attributes (like Valid_Time) make revocation and expiry straightforward.
- No central administrator needs to manually revoke access at expiry—encryption enforces it.

Summary:
In this scenario, attribute-based encryption delivers robust, flexible, and self-enforcing access control for university documents stored in the cloud, ensuring sensitive data is accessible only under precise, well-defined conditions.

4.3. Pseudo Code of Algorithm

Input: Encrypted Document C, Metadata M,

User's Attribute Keys AK, Current Time T_curr

Output: Decrypted Document D (if authorized)

1. Parse metadata M to extract:

   P, T_start, T_end, SK_enc

2. Verify time validity:

If T_curr ∉ [T_start, T_end] → Abort

3. Decrypt SK_enc using user's attribute keys:

SK_T = ABE_Decrypt (SK_enc, AK)

4. Decrypt document using SK_T:

D = Symmetric Decrypt (C, SK_T)

Return: D (only if time + attributes are valid)

4.4. Proposed Algorithm: Time-Aware ABE Encryption

```
python
CopyEdit
Algorithm: Time-Aware Attribute-Based Encryption (T-ABE)

Input: Document D, Access_Policy P (Attributes + Time), Public Key PK,
Master Key MK
Output: Encrypted Document C, Metadata M (in XML format)

1. Generate a unique session key SK
2. Embed time component T_curr into SK → SK_T = Hash(SK || T_curr)

3. Encrypt Document:
     C = Symmetric_Encrypt(D, SK_T)

4. Encrypt Session Key using ABE:
     SK_enc = ABE_Encrypt(SK_T, P, PK)

5. Create Metadata File M (XML format):
     - Store Access_Policy P
     - Store Encrypted Key SK_enc
     - Include time validity: T_start, T_end
     - Mark version or revocation token

6. Upload {C, M} to cloud

Return: {Encrypted Document C, Metadata M}
```

4.5. Decryption Phase (User Side)

```
python
CopyEdit
Algorithm: T-ABE Decryption

Input: Encrypted Document C, Metadata M, Attribute Keys (AK), Time T_curr
Output: Decrypted Document D

1. Parse XML metadata M → Extract P, T_start, T_end, SK_enc

2. Verify:
     If T_curr ∈ [T_start, T_end] → Proceed
     Else → Deny Access
```

```
3. ABE_Decrypt(SK_enc, AK) → Get SK_T

4. Symmetric_Decrypt(C, SK_T) → D

Return: Decrypted Document D
```

4.6. Key Benefits of the Proposed Framework

Feature	Benefit
Time-Varying Encryption	Reduces risk of static key leakage
Attribute-Based Control	Fine-grained and flexible access
Metadata Decoupling	Modular policy updates and revocation
XML Format	Interoperability and clarity
Cloud Adapted	Ideal for untrusted/distributed environments

Comparative Chart Showing the Existing Methods With The Proposed Method

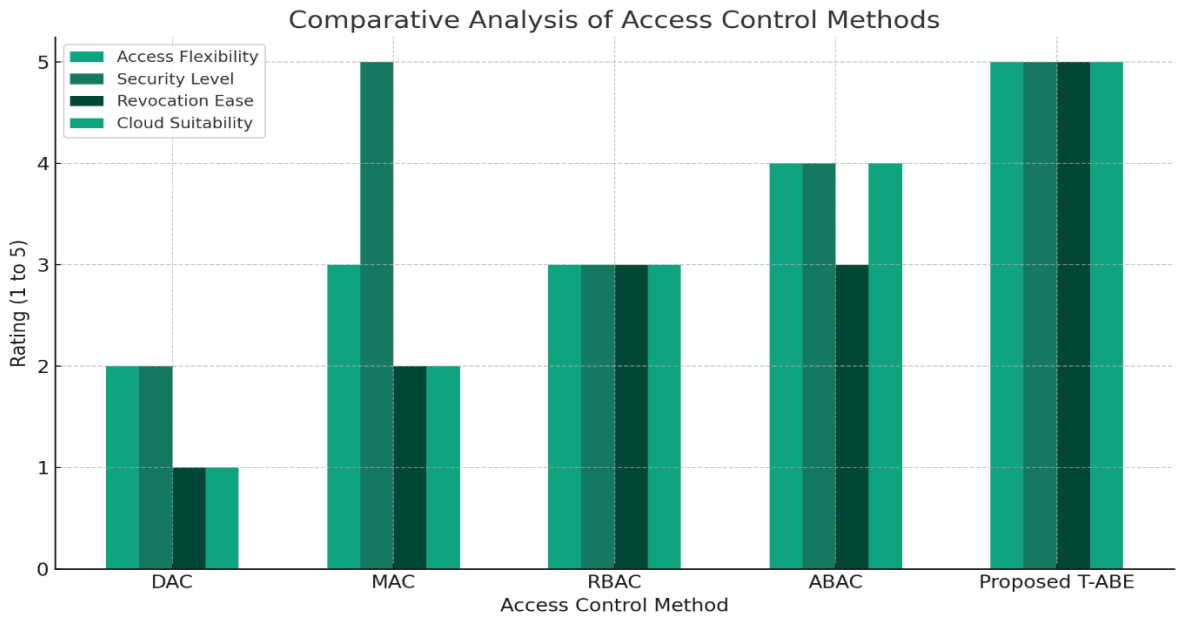
Feature / Method	**DAC (Discretionary)**	**MAC (Mandatory)**	**RBAC (Role-Based)**	**ABAC (Traditional)**	**Proposed T-ABE**
Access Control Type	Owner-controlled	System-enforced	Role-dependent	Attribute-based	Attribute + Time
Granularity	Coarse	Fine	Moderate	Fine	**Fine + Temporal**
Time-based Access	✗	✗	✗	✗	☑
Revocation Handling	Manual	Complex	Moderate	Moderate	**Easy (via XML)**
Cloud Suitability	Low	Low	Medium	High	**Very High**
Metadata Separation	✗	✗	✗	✗	☑ (XML file)
Key Leakage Resistance	Low	High	Moderate	Moderate	**High (Time-Varying)**

For Comparison the following **metrics** for each method is considered

- **Access Flexibility**
- **Security Level**
- **Revocation Ease**
- **Cloud Suitability**

Each is rated on a scale from 1 to 5.

Method	Access Flexibility	Security Level	Revocation Ease	Cloud Suitability
DAC	2	2	1	1
MAC	3	5	2	2
RBAC	3	3	3	3
ABAC	4	4	3	4
T-ABE	**5**	**5**	**5**	**5**



Graph 3: Comparison of Access Control Methods with Proposed Method

This chart compares various traditional access control mechanisms (DAC, MAC, RBAC, ABAC) with the proposed Time-Aware Attribute-Based Encryption (T-ABE) scheme. The comparison is based on Access Flexibility, Security Level, Revocation Ease, and Cloud Suitability, rated on a scale from 1 to 5.

5. Conclusion

This paper presents a comprehensive survey of Attribute-Based Encryption (ABE), outlining its mathematical foundation, including bilinear maps and the underlying security assumptions

[1, 2, 5]. A broad classification of ABE schemes is discussed along with their corresponding algorithms and limitations [3, 6, 12]. Furthermore, traditional access control models used in cloud computing—Discretionary Access Control (DAC), Mandatory Access Control (MAC), Role-Based Access Control (RBAC), and Attribute-Based Access Control (ABAC)—are reviewed with illustrative figures and comparative analysis (see Table 1 and Table 2) [7, 8, 13, 22, 23].

The proposed work introduces a modified ABE scheme integrating a time-varying encryption approach under the ABAC framework [12, 19, 27, 31]. Access permissions are managed using an XML-based metadata file, ensuring secure and dynamic control over data access [20, 22]. The future scope includes evaluating the proposed model in terms of access policy update efficiency, specifically measuring the time required to modify permissions compared to existing ABAC implementations [23, 31].

Key Contributions

The key contributions of this work are as follows:

1. **Comprehensive Survey of ABE Schemes:** We provide an in-depth review of Attribute-Based Encryption (ABE) schemes, covering their mathematical foundations, including bilinear maps and security assumptions, along with a comparative analysis of existing models [1–3, 5, 6].
2. **Integration of Access Control Models:** The paper evaluates classical access control mechanisms—Discretionary Access Control (DAC), Mandatory Access Control (MAC), and Role-Based Access Control (RBAC)—and contrasts them with Attribute-Based Access Control (ABAC) to highlight their suitability and limitations in cloud environments [7, 8, 13, 22, 23].
3. **Proposed Time-Varying ABE Framework:** We introduce a novel modification of ABE by incorporating time-varying encryption, strengthening resilience against static key exposure and unauthorized access in distributed cloud environments [12, 19, 27, 31].
4. **Policy-Decoupled Metadata Management:** Access permissions are managed separately through an XML-based metadata file, enabling dynamic policy updates, efficient revocation handling, and modular integration with ABAC-based systems [20, 22, 23].
5. **Future-Oriented Evaluation Metrics:** We outline future directions for evaluating the proposed scheme, focusing on efficiency of access policy updates, revocation costs, and scalability compared with existing revocable ABE and ABAC implementations [19, 23, 31].

Acknowledgements

We are very much thankful to all the people who helped us with this research. Also we are very much thankful to our friends and family.

References

1. Sahai A, Waters B. Fuzzy identity-based encryption. In: Cramer R (ed) *Advances in Cryptology – EUROCRYPT 2005*. Springer; 2005. p. 457–473. SpringerOpen

2. Goyal V, Pandey O, Sahai A, Waters B. Attribute-based encryption for fine-grained access control of encrypted data. *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS)*. 2006:89–98. doi:10.1145/1180405.1180418. SpringerOpen
3. Bethencourt J, Sahai A, Waters B. Ciphertext-policy attribute-based encryption. *IEEE Symposium on Security and Privacy*. 2007:321–334. doi:10.1109/SP.2007.11. SpringerOpen
4. Ostrovsky R, Sahai A, Waters B. Attribute-based encryption with non-monotonic access structures. *ACM CCS*. 2007:195–203. doi:10.1145/1315245.1315270. SpringerOpen
5. Waters B. Ciphertext-Policy Attribute-Based Encryption: An expressive, efficient, and provably secure realization. In: *PKC 2011*. Springer; 2011. p. 53–70. doi:10.1007/978-3-642-20465-4_24. su.diva-portal.org
6. Lewko A, Waters B. Decentralizing attribute-based encryption. In: *EUROCRYPT 2011*. Springer; 2011. p. 568–588. doi:10.1007/978-3-642-20465-4_33. IACR
7. Green M, Hohenberger S, Waters B. Outsourcing the decryption of ABE ciphertexts. *USENIX Security*. 2011. (Open access). static.usenix.orgUSENIX
8. NIST. *Attribute Based Access Control (ABAC) Definition and Considerations*. NIST SP 800-162. 2014. NIST PublicationsNIST Computer Security Resource Center
9. Wan Z, Liu J, Deng RH. HASBE: A hierarchical attribute-based solution for flexible and scalable access control in cloud computing. *IEEE Trans Inf Forensics Security*. 2012;7(2):743–754. doi:10.1109/TIFS.2011.2172209. ACM Digital Library
10. Wang S, Zhou J, Liu JK, Yu J, Chen J, Xie W. An efficient file hierarchy attribute-based encryption scheme in cloud computing. *IEEE Trans Inf Forensics Security*. 2016;11(6):1265–1277. doi:10.1109/TIFS.2016.2523941. SpringerOpen
11. Wang S, Liang K, Liu JK, Chen J, Wong DS, Xie W. Attribute-based data sharing scheme revisited in cloud computing. *IEEE Trans Inf Forensics Security*. 2016;11(8):1661–1673. doi:10.1109/TIFS.2016.2549004. SpringerOpen
12. Poster: Temporal attribute-based encryption in clouds (TABE). *ACM CCS*. 2011. doi:10.1145/2046707.2093517. ACM Digital Library+1
13. Zhou L, Varadharajan V, Hitchens M. Achieving secure role-based access control on encrypted data in cloud storage. *IEEE Trans Inf Forensics Security*. 2013;8(12):1947–1960. (context to RBAC/ABE).
14. Yang K, Jia X, Ren K, et al. DAC-MACs: Effective data access control for multiauthority cloud storage systems. *IEEE Trans Inf Forensics Security*. 2013;8(11):1790–1801.
15. Xu S, Yuan J, Xu G, Wang J. Efficient CP-ABE with black-box traceability. *Information Sciences*. 2020;538:19–38. doi:10.1016/j.ins.2020.05.115. SpringerOpen
16. Liu Z, Cao Z, Wong DS. White-box traceable CP-ABE supporting any monotone access structures. *IEEE Trans Inf Forensics Security*. 2013;8(1):76–88.
17. Ning J, Dong X, Cao Z, Wei L, Lin X. White-box traceable CP-ABE supporting flexible attributes. *IEEE Trans Inf Forensics Security*. 2015;10(6):1274–1288.
18. Wan Z, Liu J, Deng RH. Enforcing access control in virtual organizations using hierarchical ABE. *arXiv:1205.5757*. 2012. arXiv
19. Qin B, Zhao Q, Zheng D, et al. (Dual) server-aided revocable ABE with decryption key exposure resistance. *Information Sciences*. 2019;490:74–92. doi:10.1016/j.ins.2019.03.053. SpringerOpen
20. Cui H, Yuen TH, Deng RH, Wang G. Server-aided revocable attribute-based encryption. *ESORICS 2016*. Springer; 2016. (chapter; model introduction). InKCORE

21. Wang H, Li Y, Susilo W, et al. Fast and flexible attribute-based searchable encryption with multi-search. *Computer Standards & Interfaces*. 2022;82:103635. doi:10.1016/j.csi.2022.103635. SpringerOpen
22. Xue Y, Xue K, Gai N, et al. Attribute-based controlled collaborative access control for public cloud storage. *IEEE Trans Inf Forensics Security*. 2019;14(11):2927–2942. doi:10.1109/TIFS.2019.2911166. SpringerOpen
23. Li X, Wang H, Ma S, Xiao M, Huang Q. Revocable and verifiable weighted ABE with collaborative access for EHR in cloud. *Cybersecurity*. 2024;7:18. doi:10.1186/s42400-024-00211-1. (Your target journal; strong 2024 addition.) SpringerOpen
24. Alasmary W, Chung S, Matrawy A. Attribute-based encryption for cloud access control: A survey. *ACM Computing Surveys*. 2020;53(4):1–41. doi:10.1145/3398036. ACM Digital Library
25. Lai J, Guo F, Susilo W, et al. Generic conversions from CPA to CCA for threshold ABE with constant-size ciphertexts. *Information Sciences*. 2023;613:966–981. doi:10.1016/j.ins.2022.08.069. SpringerOpen
26. Huang X, Xiong H, Chen J, et al. Efficient revocable storage ABE with arithmetic span programs in cloud-assisted IoT. *IEEE Trans Cloud Comput*. 2023;11(2):1273–1285. doi:10.1109/TCC.2021.3131686. SpringerOpen
27. Wei J, Chen X, Huang X, et al. RS-HABE: Revocable-storage & hierarchical ABE for secure e-health sharing in public cloud. *IEEE Trans Dependable and Secure Computing*. 2021;18(5):2301–2315. doi:10.1109/TDSC.2019.2947920. SpringerOpen
28. Hoang VH, Lehtihet E, Ghamri-Doudane Y. Forward-secure data outsourcing based on revocable ABE. *IWCMC 2019*. doi:10.1109/IWCMC.2019.8766674. SpringerOpen
29. Sun Z, Zhang Y, Zhang S, et al. Outsourced CP-ABE with partial decryption and verifiability. *International Journal of Distributed Sensor Networks*. 2020;16(6). doi:10.1177/1550147720926368. (On outsourced decryption verifiability.) SAGE Journals
30. Green M, Hohenberger S, Waters B. (context) Online/offline ABE and follow-ups. In: *PKC 2014* (surveyed). Springer; 2014. (Background linkage.) IACRSpringerLink
31. **Latest**: Karuppiah R, Sangeetha K, et al. Effective forward-secure key-policy ABE with ciphertext delegation for cloud storage. *Information Sciences*. 2025 (in press). Elsevier Article ID: S0020025525006917. (Very recent forward-secure KP-ABE). ScienceDirect
32. **Latest**: Cai D, Chen B, Zhang L, Li K, Kan H. Attribute-Based Encryption with payable outsourced decryption using blockchain and responsive zero-knowledge proof. *arXiv:2411.03844*. 2024. (Practical payable/verifiable outsourcing). arXiv
33. Survey (revocation focus): Chen X, Li J, Liang K, et al. Survey on revocation in CP-ABE. *Applied Sciences*. 2019;9(5):909 (open-access overview). PMC
34. Qin B, Deng RH, Li Y, Liu S. Server-aided revocable IBE/ABE: models and constructions (multi-user secure SR-ABE). In: *Computer Security – ESORICS 2021*. Springer; 2021. (Multi-user-secure SR-ABE). ACM Digital Library
35. Wang H, Ni W, Liu D, et al. Unified ciphertext-policy weighted ABE for data sharing in cloud computing. *Applied Sciences*. 2018;8(12):2519. doi:10.3390/app8122519. SpringerOpen
36. Wang Y, Zhang D, Zhong H. Multi-authority weighted ABE in cloud computing. *ICNC 2014*. doi:10.1109/ICNC.2014.6975982. SpringerOpen
37. Merkle RC. Protocols for public key cryptosystems. *IEEE Symposium on Security and Privacy*. 1980:122–134. doi:10.1109/SP.1980.10006. (Classic building block). SpringerOpen