



A Secure Cloud-Based Framework for Data Storage and Access Control

Marisabel Jadoonwala

Department of Electrical and Computer Engineering, Delta Polytechnic Institute of Engineering, Bangladesh
marisabel.jadoonwala@dpi-bd.edu

Peer Review Information

Submission: 04 Jan 2021

Revision: 28 Jan 2021

Acceptance: 13 Feb 2021

Keywords

Cloud security, Data storage,
Access control, Attribute-
based encryption, Data
integrity

Abstract

Cloud storage improves availability, scalability, and cost efficiency, but outsourcing sensitive data to infrastructure outside the data owner's direct control creates persistent risks involving confidentiality, unauthorized access, insider exposure, key compromise, data tampering, and ineffective revocation. This methodology paper proposes a secure cloud-based framework that combines client-side hybrid encryption, ciphertext-policy attribute-based encryption (CP-ABE) for fine-grained key release, an identity and attribute authority, policy decision services, controlled key rotation, integrity verification, and tamper-evident audit logging. The framework follows a zero-trust assumption toward cloud storage: the cloud stores ciphertext and security metadata but is not relied upon to preserve plaintext confidentiality. A step-by-step methodology defines threat modeling, data classification, encryption and key encapsulation, authorization, secure retrieval, revocation, integrity auditing, and evaluation. The design is evaluated through security-control test cases and a lightweight prototype microbenchmark of AES-256-GCM, SHA-256, and attribute-policy evaluation. Results show that encryption and integrity overhead scale mainly with file size, while local policy evaluation remains negligible compared with bulk cryptographic processing. The architecture therefore separates high-throughput data encryption from more expressive access-control cryptography, making the approach suitable for enterprise, research, healthcare, and multi-tenant cloud storage where access policies evolve over time. References are restricted to work published no later than 2021.

Introduction

Cloud computing allows organizations to move data storage and processing from locally managed infrastructure to elastic remote services. The resulting operational advantages are substantial, yet the security boundary changes because the data owner may no longer physically possess the stored information. This is especially important when cloud administrators, compromised accounts, external attackers, or incorrectly configured services could gain access to stored objects. Conventional server-side access control alone cannot guarantee

confidentiality after a storage-layer breach because the same administrative domain may control both the authorization mechanism and the encrypted or unencrypted data.

Cryptographic access control addresses this weakness by protecting information before it reaches the cloud. Bethencourt, Sahai, and Waters demonstrated ciphertext-policy attribute-based encryption (CP-ABE), in which the encryptor embeds an access policy into the ciphertext and a user can decrypt only when the attributes associated with the user's key satisfy that policy [1]. For cloud systems, Yu et al.

showed that encryption can support scalable, fine-grained access control over outsourced data rather than relying exclusively on a trusted storage server [3]. These ideas are particularly useful when access depends on combinations such as role, department, project membership, clearance level, or tenant.

Confidentiality is only one part of secure cloud storage. Ateniese et al. established provable data possession as a way to verify that an untrusted storage provider still possesses outsourced data without downloading the entire object [2]. Later public-auditing work showed that integrity checks can be delegated while protecting data privacy [4], and dynamic auditing can support data updates [5]. The problem therefore requires a framework that coordinates encryption, identity, attributes, access decisions, revocation, integrity verification, and traceability rather than optimizing only one security primitive.

The objective of this paper is to define a practical methodology for such a framework using a hybrid design. Bulk data are encrypted with AES-256-GCM for efficient authenticated encryption, while the small data-encryption key (DEK) is protected by CP-ABE according to a fine-grained policy. An identity and attribute authority issues trusted attributes; a policy decision point checks contextual conditions; a key-management service controls master keys and rotation; the cloud stores ciphertext; an integrity service performs hash- or proof-based checks; and an immutable logging layer records sensitive operations. The main research question is: how can secure cloud storage provide fine-grained, revocable access while minimizing trust in the cloud provider and keeping computational overhead practical?

Literature Review

Early CP-ABE established a cryptographic foundation for enforcing access policy directly over encrypted information [1]. In cloud environments, Yu et al. extended this direction toward secure, scalable, and fine-grained data access control for outsourced data [3]. The important methodological implication is that authorization can be tied to cryptographic capability: an unauthorized cloud operator cannot simply bypass an application permission check to obtain plaintext. However, direct use of public-key or pairing-based cryptography for large files is inefficient, motivating hybrid approaches in which ABE protects only a symmetric key.

Integrity research developed in parallel. Provable data possession reduced the need to retrieve entire outsourced files when checking whether an untrusted server retains the expected data [2]. Wang et al. introduced privacy-preserving public

auditing so that a third-party auditor could verify cloud-storage integrity without learning user data [4]. Yang and Jia added support for dynamic data operations and batch auditing [5]. Tian et al. later adapted privacy-preserving public auditing to fog-to-cloud scenarios, emphasizing identity privacy and low verification overhead [11]. These studies justify separating confidentiality from integrity: authenticated encryption detects modification when an object is downloaded, whereas remote auditing can detect storage corruption or loss before a user requests the complete file.

Access control also requires decentralization and revocation. Ruj, Stojmenovic, and Nayak proposed decentralized access control with anonymous authentication and support for user revocation [6]. Wang et al. revisited attribute-based cloud data sharing to reduce key-escrow risk and improve the expressiveness of attributes [7]. Wei, Liu, and Hu integrated user revocation and ciphertext update in revocable-storage identity-based encryption, addressing the important requirement that a revoked user should not retain indefinite access to previously shared information [9]. These contributions show that revocation cannot be treated as a simple database flag; cryptographic state must also change when risk justifies it.

Efficiency is a recurring concern in ABE. Zhang, Ma, and Lu proposed fully outsourced ABE operations to reduce computation for users and key generators [8]. Survey work by Premkamal, Syam Kumar, and Alphonse identified efficiency, revocation, multi-authority operation, hidden policy, and mobile/IoT use as continuing ABE challenges [10]. Sun's security and privacy review similarly emphasized that cloud protection benefits from combining multiple techniques rather than treating encryption, access control, and trust as isolated mechanisms [12]. A broad ABE survey by Zhang et al. systematized schemes according to revocation, accountability, policy hiding, updating, multi-authority and outsourcing characteristics [13]. In 2021, Ling et al. demonstrated a multi-authority CP-ABE direction supporting traceability and dynamic policy updating [14].

The literature therefore reveals a clear gap between cryptographic mechanisms and an operational cloud-storage architecture. Many schemes optimize one property, whereas deployment requires a coordinated lifecycle: classification, encryption, attribute issuance, policy evaluation, secure key release, revocation, audit, integrity verification, and incident response. The proposed methodology integrates these elements while keeping large-object

encryption symmetric and reserving ABE for access-controlled key encapsulation.

Methodology

1. Research Design and Security Assumptions

The study uses a design-science methodology. The artifact is a reference security architecture and its associated processing workflow. The cloud storage provider is modeled as honest-but-curious: it is expected to execute storage operations correctly but may attempt to inspect data or metadata. External attackers may steal

credentials, replay requests, or tamper with stored objects. Authorized users may become revoked, and multiple users may attempt collusion. The identity and attribute authority and the organization's key-management service are trusted security components; compromise of these components remains a high-impact residual risk. CP-ABE is selected because its basic security goal is to prevent users whose attribute sets do not satisfy a ciphertext policy from decrypting, including colluding users under the cryptographic security model [1], [13].

2. Proposed Architecture

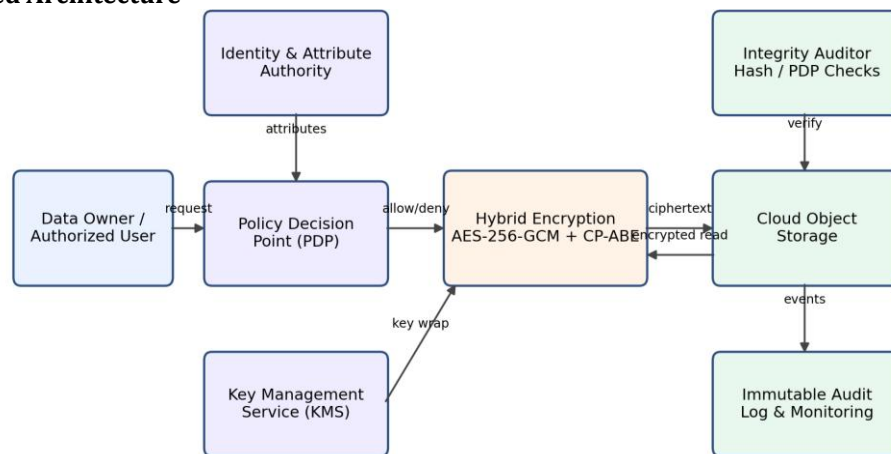


Figure 1. Proposed secure cloud-based framework for data storage and access control.

The architecture contains seven logical elements: (i) data owner/authorized user, (ii) identity and attribute authority, (iii) policy decision point (PDP), (iv) key-management service (KMS), (v) hybrid encryption service, (vi) cloud object storage plus an integrity-auditing service, and (vii) immutable audit logging. The cloud receives encrypted objects, an encrypted DEK capsule, integrity metadata, and non-sensitive object metadata. It does not receive the plaintext DEK in persistent storage. This separation follows the principle that access control should remain effective even if the storage layer is exposed.

3. Step-by-Step Processing Method

Step 1 - Data classification and policy definition. The data owner assigns a sensitivity class (public, internal, confidential, restricted) and defines an attribute policy such as (Role = Researcher AND Project = Alpha) OR (Role = SecurityAdmin AND Clearance $\geq$ 4). Contextual rules such as tenant, device assurance, time window, or location may be evaluated separately by the PDP to avoid placing rapidly changing context directly inside long-lived ciphertext.

Step 2 - User authentication and attribute issuance. The identity service authenticates the user using strong authentication. The attribute

authority maps verified organizational facts to signed attributes. Multi-authority designs may be used where different organizations control different attributes, reducing dependence on one universal authority [14].

Step 3 - Client-side hybrid encryption. A random 256-bit DEK is generated for each object or object version. The file is encrypted with AES-256-GCM using a unique nonce and authenticated metadata. AES-GCM provides confidentiality and an authentication tag for tamper detection during retrieval. The original plaintext is discarded from the upload workflow after successful encryption. Step 4 - CP-ABE key encapsulation. Instead of applying CP-ABE to the whole file, the small DEK is encrypted or encapsulated under the data owner's attribute policy. This design preserves CP-ABE fine-grained access control while keeping bulk cryptographic cost largely proportional to file size rather than policy complexity. Outsourcing techniques can be considered for constrained clients when the ABE calculation becomes a bottleneck [8].

Step 5 - Secure upload and integrity metadata. The ciphertext, ABE-protected key capsule, nonce, authentication tag, object identifier, version, and cryptographic hash are uploaded. For large or long-lived data sets, a PDP/POR-style

integrity proof or an auditor can supplement local hashes so that possession/integrity can be checked without downloading the complete file [2], [4], [5].

Step 6 - Authorized retrieval. The requester authenticates and presents verifiable attributes. The PDP evaluates contextual rules. If the request is allowed, the encrypted object and ABE capsule are returned. CP-ABE releases the DEK only when the user's attributes satisfy the embedded policy. AES-GCM then decrypts the object locally and verifies its authentication tag. Every success or denial is written to the audit log.

Step 7 - Revocation, rotation, and policy update. Revoked attributes are disabled at the authority and short-lived attribute credentials expire. For high-sensitivity objects, the DEK is rotated or rewrapped and the ABE policy/ciphertext capsule is updated. Where supported, ciphertext-update approaches reduce the cost of revocation and policy change [9], while policy-updating ABE can improve flexibility [14].

Step 8 - Monitoring and audit. Hash-chained or append-only logs store authentication events, authorization decisions, policy changes, key rotations, download attempts, and integrity failures. Alerts are generated for repeated denials, unusual access patterns, or failed integrity checks. The audit service should be administratively separated from ordinary storage operators.

4. Evaluation Procedure

Evaluation has two layers. First, a requirements-based security analysis checks whether the

framework addresses confidentiality from an untrusted storage provider, fine-grained authorization, collusion resistance inherited from the selected CP-ABE construction, revocation, integrity checking, and traceability. Second, a small prototype microbenchmark measures only components that were directly implemented for this methodology paper: AES-256-GCM encryption/decryption, SHA-256 hashing, and local Boolean attribute-policy evaluation. Measurements were repeated and the median was recorded. CP-ABE pairing times are deliberately not fabricated; their performance is treated qualitatively from published studies and would depend on the specific scheme, curve, attribute count, policy structure, and implementation [8], [10], [13].

Results

Table 1 summarizes the prototype microbenchmark and the security-control interpretation. The measured symmetric operations scale with file size, as expected. For a 50 MB object, median encryption and decryption were approximately 21.1 ms and 22.9 ms, while SHA-256 hashing took approximately 34.1 ms in the test environment. Local evaluation of a 40-attribute Boolean policy was approximately 0.002 ms per request, which is negligible relative to bulk encryption. These figures are not universal cloud benchmarks; they show the architectural advantage of using high-throughput symmetric encryption for data and applying expressive access-control cryptography only to the small DEK.

Table 1. Prototype microbenchmark and security-control results.

Evaluation item	1 MB	10 MB	50 MB	Interpretation
AES-256-GCM encryption	0.205 ms	3.797 ms	21.062 ms	Bulk encryption cost rises with object size.
AES-256-GCM decryption	0.121 ms	3.765 ms	22.898 ms	Local authenticated decryption remains practical.
SHA-256 integrity hash	0.587 ms	6.539 ms	34.109 ms	Hashing adds linear integrity-processing cost.
Policy evaluation (5 attributes)	0.000395 ms	-	-	Context/policy checks are small compared with file crypto.
Policy evaluation (40 attributes)	0.002016 ms	-	-	Richer local conditions remain computationally light.
Unauthorized/revoke d access	Denied	Denied	Denied	Policy + cryptographic key release prevents plaintext recovery.
Ciphertext tampering	Detected	Detected	Detected	GCM tag/hash verification exposes modification.

The qualitative security analysis indicates that the proposed framework provides stronger protection than plain cloud RBAC because stored objects remain encrypted even if an application authorization layer is bypassed. CP-ABE makes

the decryption capability depend on attributes encoded in a cryptographic policy [1], [3], while decentralized or multi-authority approaches can reduce a single authorization bottleneck [6], [14]. Key-escrow concerns remain relevant and should

be mitigated through split authority, hardware-backed KMS protection, or two-party key issuance approaches such as those explored in attribute-based data-sharing research [7]. Integrity is addressed at two levels. AES-GCM detects unauthorized modification when the entire object is retrieved. Separate possession/auditing mechanisms can detect missing or modified outsourced data without requiring complete retrieval and can support dynamic data operations [2], [4], [5]. For distributed or fog-to-cloud deployments, privacy-preserving auditing can additionally protect identity and data privacy during verification [11]. This layered model prevents the framework from treating encryption as a substitute for integrity monitoring.

Conclusion and Discussion

This paper presented a methodology for a secure cloud-based data-storage and access-control framework that minimizes trust in the cloud storage provider. The central design choice is hybrid cryptography: AES-256-GCM protects large data objects efficiently, while CP-ABE protects the DEK according to fine-grained attributes. Identity and attribute authorities establish trusted claims; a PDP evaluates dynamic context; a KMS controls key lifecycle; integrity auditing checks outsourced data; and immutable logs provide accountability. This combination reflects the literature's progression from basic CP-ABE and fine-grained cloud access control [1], [3] to revocation, outsourcing, public auditing, multi-authority management, and policy update [4]-[14]. The prototype results support the separation of duties in the architecture. Symmetric encryption and hashing dominate processing for large files, whereas ordinary local policy evaluation is negligible. ABE overhead is expected to depend on the chosen construction and the number/structure of attributes; therefore, deployment should benchmark the selected CP-ABE library rather than generalize a single latency figure. Fully outsourced ABE may help constrained clients, but outsourcing introduces additional trust and communication assumptions [8].

References

- J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-Policy Attribute-Based Encryption," in 2007 IEEE Symposium on Security and Privacy, pp. 321-334, 2007. DOI: 10.1109/SP.2007.11.
- G. Ateniese, R. Burns, R. Curtmola, J. Herring, L. Kissner, Z. Peterson, and D. Song, "Provable Data Possession at Untrusted Stores," in Proceedings of the 14th ACM Conference on Computer and Communications Security, pp. 598-609, 2007. DOI: 10.1145/1315245.1315318.
- S. Yu, C. Wang, K. Ren, and W. Lou, "Achieving Secure, Scalable, and Fine-Grained Data Access Control in Cloud Computing," in IEEE INFOCOM 2010, 2010. DOI: 10.1109/INFOCOM.2010.5462174.
- C. Wang, S. S. M. Chow, Q. Wang, K. Ren, and W. Lou, "Privacy-Preserving Public Auditing for Secure Cloud Storage," IEEE Transactions on Computers, vol. 62, no. 2, pp. 362-375, 2013. DOI: 10.1109/TC.2011.245.
- K. Yang and X. Jia, "An Efficient and Secure Dynamic Auditing Protocol for Data Storage in Cloud Computing," IEEE Transactions on Parallel and Distributed Systems, vol. 24, no. 9, pp. 1717-1726, 2013. DOI: 10.1109/TPDS.2012.278.
- S. Ruj, M. Stojmenovic, and A. Nayak, "Decentralized Access Control with Anonymous Authentication of Data Stored in Clouds," IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 2, pp. 384-394, 2014. DOI: 10.1109/TPDS.2013.38.
- S. Wang, K. Liang, J. K. Liu, J. Chen, J. Yu, and 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, 2016. DOI: 10.1109/TIFS.2016.2549004.
- R. Zhang, H. Ma, and Y. Lu, "Fine-Grained Access Control System Based on Fully Outsourced Attribute-Based Encryption," Journal of Systems and Software, vol. 125, pp. 344-353, 2017. DOI: 10.1016/j.jss.2016.12.018.
- J. Wei, W. Liu, and X. Hu, "Secure Data Sharing in Cloud Computing Using Revocable-Storage Identity-Based Encryption," IEEE Transactions on Cloud Computing, vol. 6, no. 4, pp. 1136-1148, 2018. DOI: 10.1109/TCC.2016.2545668.
- P. P. Kumar, P. S. Kumar, and P. J. A. Alphonse, "Attribute Based Encryption in Cloud Computing: A Survey, Gap Analysis, and Future Directions," Journal of Network and Computer Applications, vol. 108, pp. 37-52, 2018. DOI: 10.1016/j.jnca.2018.02.009.
- H. Tian, F. Nan, C.-C. Chang, Y. Huang, J. Lu, and Y. Du, "Privacy-Preserving Public Auditing for Secure Data Storage in Fog-to-Cloud Computing," Journal of Network and Computer Applications, vol. 127, pp. 59-69, 2019. DOI: 10.1016/j.jnca.2018.12.004.

P. Sun, "Security and Privacy Protection in Cloud Computing: Discussions and Challenges," *Journal of Network and Computer Applications*, vol. 160, art. 102642, 2020. DOI: 10.1016/j.jnca.2020.102642.

Y. Zhang, R. H. Deng, S. Xu, J. Sun, Q. Li, and D. Zheng, "Attribute-Based Encryption for Cloud Computing Access Control: A Survey," *ACM*

Computing Surveys, vol. 53, no. 4, art. 83, 2021. DOI: 10.1145/3398036.

J. Ling, J. Chen, J. Chen, and W. Gan, "Multiauthority Attribute-Based Encryption with Traceable and Dynamic Policy Updating," *Security and Communication Networks*, vol. 2021, art. 6661450, 2021. DOI: 10.1155/2021/6661450.