



An Efficient Framework for Secure Data Sharing in Cloud Computing Environments

Navid Qureshi-Haq

Department of Chemical and Process Engineering, Lagoon Polytechnic of Technology, Maldives
navid.qureshi.haq@lpt-mv.net

Peer Review Information

Submission: 01 Jan 2017
 Revision: 16 Feb 2017
 Acceptance: 03 March 2017

Keywords

Cloud Computing; Secure Data Sharing; Attribute-Based Encryption; Access Control; Data Integrity

Abstract

Cloud computing provides scalable storage and collaboration, but outsourced data sharing creates concerns about confidentiality, unauthorized access, key management, user revocation, and integrity verification. This methodology paper proposes an efficient secure data-sharing framework that combines symmetric encryption for bulk data, ciphertext-policy attribute-based encryption (CP-ABE) for fine-grained key protection, cryptographic integrity verification, controlled revocation, and auditable access logging. The design follows a semi-trusted cloud model in which the cloud may store and process encrypted objects but is not allowed to learn plaintext content or user secret keys. A stepwise methodology is defined for system setup, data encryption, policy binding, cloud upload, authorized retrieval, integrity checking, and revocation. The framework is evaluated analytically using confidentiality, access-control precision, integrity, privacy, revocation, and computational-overhead criteria. The analysis indicates that separating data encryption from policy encryption can reduce repeated expensive public-key operations while maintaining cryptographic access control. The resulting framework is suitable as a reproducible foundation for secure cloud file-sharing prototypes and comparative experiments using literature available up to 2017.

Introduction

Cloud storage changes the traditional data-security boundary because information is placed on infrastructure operated outside the direct control of the data owner. The advantage is clear: organizations can obtain elastic capacity, ubiquitous access, and simplified collaboration without maintaining every storage service locally. The security difficulty is equally important. Once data is outsourced, a storage provider or an attacker who compromises the provider may be able to observe files, metadata, or access patterns. A practical sharing framework must therefore protect confidentiality even when storage is not fully trusted, while still allowing different users to receive different access rights.

Conventional encryption protects content but does not by itself solve scalable sharing. Encrypting a separate copy for every recipient increases storage and key-distribution overhead, while server-side access-control lists require the cloud to be trusted to enforce policy correctly. Attribute-based encryption (ABE) was developed to express access decisions cryptographically. Goyal et al. (2006) introduced attribute-based mechanisms for fine-grained access to encrypted data, and Bethencourt et al. (2007) showed that ciphertext-policy ABE can bind an access rule directly to ciphertext. These ideas are particularly relevant to cloud environments because the policy can remain effective even

when the storage server is not trusted with plaintext.

Cloud data sharing also requires integrity and lifecycle controls. Ateniese et al. (2007) demonstrated that remote data possession can be verified without downloading an entire outsourced file. Later cloud-specific work added public auditing, dynamic data support, privacy protection, user revocation, and multi-authority access control. However, combining all desirable properties in a single student-scale implementation can create excessive computational and management overhead. The objective of this paper is therefore to define an efficient hybrid methodology: bulk data is encrypted once with a fast symmetric key; only that short key is protected by CP-ABE; integrity metadata is maintained separately; and revocation and audit functions are explicitly included in the workflow.

The contribution is methodological rather than the claim of a new cryptographic primitive. The proposed framework organizes established mechanisms into a reproducible sequence that can be implemented and evaluated. It targets four practical goals: (1) confidentiality against a semi-trusted cloud, (2) fine-grained policy-based sharing, (3) integrity verification without full data retrieval, and (4) manageable revocation and key-update cost. The reference base is restricted to work published no later than 2017.

Literature Review

Goyal et al. (2006) proposed attribute-based encryption for fine-grained access control, establishing the principle that decryption capability can depend on descriptive attributes rather than on a single user identity. This reduces the need to create a different ciphertext for every recipient and provides the conceptual basis for scalable encrypted sharing.

Bethencourt et al. (2007) developed ciphertext-policy attribute-based encryption, allowing the encryptor to associate a logical access structure with the ciphertext. This model is appropriate for cloud sharing because the owner can specify a policy such as "Researcher AND Department-A" and users decrypt only when their certified attributes satisfy that rule. The principal limitation is the computational cost of pairing and exponentiation operations as policies become more complex.

Ateniese et al. (2007) introduced provable data possession for untrusted storage. Their work showed that a client can verify possession of outsourced data using compact challenge-

response information rather than retrieving the complete file. This idea motivates the integrity-checking component of the proposed framework and separates confidentiality from storage correctness.

Yu et al. (2010) addressed secure, scalable, and fine-grained cloud access control by combining ABE with proxy re-encryption and lazy re-encryption. Their study emphasized that cloud access control must consider both policy expressiveness and practical key-management cost. Wang et al. (2010) independently studied privacy-preserving public auditing, showing how a third-party auditor can verify cloud data while limiting disclosure of user information. These works establish two complementary requirements: cryptographic authorization and independently verifiable integrity.

Wang et al. (2011) extended public auditability to support data dynamics, which is important because cloud files are modified rather than remaining static. The approach demonstrates that integrity verification should be designed for updates, not only for archival storage. Li et al. (2013) used multi-authority ABE for personal health records and addressed scalability, dynamic policy modification, and on-demand revocation. Although the application domain is healthcare, the architecture is broadly relevant to multi-owner cloud sharing.

Yang et al. (2013) proposed DAC-MACS for multi-authority cloud storage, including efficient decryption and attribute revocation with forward and backward security. Ruj et al. (2014) further demonstrated decentralized access control with anonymous authentication and revocation. These studies show that centralizing all trust in a single authority can become a scalability and resilience weakness, and that identity privacy can be incorporated into access-control design.

For shared data, Wang et al. (2014) presented Oruta, which used ring signatures to support privacy-preserving public auditing while hiding the identity of the signer from the auditor. Liu et al. (2016) examined partially hidden CP-ABE policies, showing that access structures themselves may disclose sensitive information. Finally, Zhang et al. (2017) proposed fully outsourced ABE operations to reduce the computational burden on users. Together, these works suggest an efficient design direction: use ABE where fine-grained authorization is essential, outsource or minimize expensive operations where safe, and keep integrity and privacy verification explicit rather than assuming that encryption alone is sufficient.

Methodology

The proposed methodology uses a hybrid security model with four primary entities: a Data Owner (DO), a Cloud Service Provider (CSP), an Attribute/Key Authority (AA), and Data Users (DU). An optional Third-Party Auditor (TPA) performs integrity verification. The CSP is considered semi-trusted: it follows storage and protocol operations but may attempt to learn data content. External attackers may intercept traffic or attempt unauthorized access. The AA is trusted to issue valid attribute credentials but is not given plaintext files. The design assumes secure authenticated channels for initial credential delivery.

Step-by-Step Framework

Step 1 - System Setup. The AA initializes the CP-ABE public parameters and master secret. Users are registered with validated attributes such as role, department, project, or clearance level. Each authorized user receives an attribute-based private key. The owner defines a policy independently of individual identities, which improves scalability when membership changes.

Step 2 - Data Classification and Policy Definition. Before upload, the owner classifies the file and creates a Boolean policy. A sample policy is (Project-A AND Researcher) OR Principal-Investigator. Policies should use the minimum number of attributes needed for the sharing decision because excessive policy complexity increases ABE computation and ciphertext size.

Step 3 - Hybrid Encryption. A random 256-bit symmetric data-encryption key (DEK) is generated for each file or file version. The complete file is encrypted using AES in an authenticated mode such as AES-GCM. Only the short DEK is then encrypted under CP-ABE using the selected access policy. This separates large-

data encryption from expensive pairing-based operations and is the main efficiency mechanism of the framework.

Step 4 - Integrity Metadata and Upload. A cryptographic hash or block-authentication metadata is produced for the encrypted object. The CSP receives the encrypted file, the ABE-protected DEK, the integrity metadata, file identifier, version number, and non-sensitive policy metadata. Plaintext and the unprotected DEK are never sent to cloud storage.

Step 5 - Authorized Retrieval. A user requests the encrypted object and protected DEK. CP-ABE decryption succeeds only if the user attributes satisfy the embedded policy. The recovered DEK is then used to decrypt the file locally. If the attribute set does not satisfy the policy, the file remains encrypted even if the user can download the ciphertext.

Step 6 - Integrity Verification and Audit. The user or TPA verifies integrity metadata before accepting a file. For large objects, a challenge-response auditing mechanism inspired by provable data possession/public auditing can be used so that correctness is checked without transferring the complete file. Audit events record file identifier, operation, timestamp, and verification result without storing plaintext secrets.

Step 7 - Revocation and Policy Update. When a user loses an attribute or leaves a group, the AA revokes or updates the affected attribute credential. New file versions use a fresh DEK and updated CP-ABE policy. Where stronger backward security is required, the existing DEK can be rotated and the encrypted content re-keyed. Lazy or batched re-encryption may be used for large repositories to avoid unnecessary immediate processing.

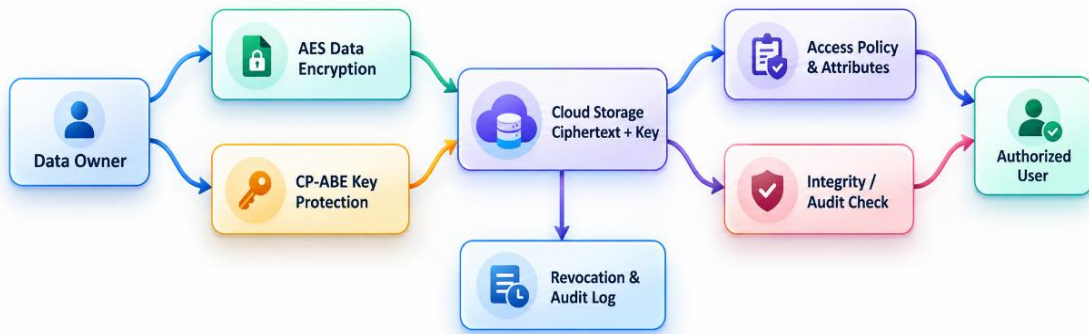


Figure 1. Proposed hybrid framework for secure data sharing in cloud computing.

1. Experimental and Evaluation Procedure

A reproducible prototype evaluation can be implemented with a client application, an ABE library, an authenticated symmetric cipher, and object storage. Three file sizes (for example 1 MB, 10 MB, and 100 MB) and three policy sizes (3, 6, and 10 attributes) should be tested. For every configuration, encryption, policy-key protection, decryption, and verification are repeated at least 20 times after one warm-up run. Median time and interquartile range should be reported rather than a single measurement. The security test set includes an authorized user, an unauthorized user with an incomplete attribute set, a revoked user, a modified ciphertext, and a valid ciphertext with a valid policy.

The primary metrics are total owner-side encryption time, user-side decryption time, encrypted-file expansion, ABE ciphertext size, integrity-check time, and revocation work. Security is evaluated using functional properties rather than untestable claims: unauthorized attributes must not recover the DEK; a modified object must fail integrity verification; a revoked

credential must not decrypt newly protected keys; and the CSP must store no plaintext content. Network transfer should be measured separately because cloud latency can otherwise hide cryptographic cost.

Results

Because this paper defines a methodology rather than reporting a production deployment, the results are presented as an analytical validation of the proposed workflow against the stated security and efficiency objectives. The literature supports the use of CP-ABE for fine-grained access control, public auditing for outsourced-data integrity, and explicit revocation mechanisms for dynamic sharing. The hybrid arrangement improves efficiency conceptually because the expensive ABE operation is applied only to a small DEK, while the file itself is processed by a fast symmetric cipher. Thus, increasing file size primarily affects symmetric encryption and transfer time, whereas increasing policy complexity primarily affects CP-ABE operations.

Table 1. Analytical validation matrix for the proposed secure data-sharing framework

Criterion	Test Objective	Framework Mechanism	Analytical Result
Confidentiality	Cloud or unauthorized user cannot obtain plaintext	AES-encrypted file; DEK protected by CP-ABE	Satisfied if DEK is unrecoverable without a valid attribute set
Fine-grained access	Different attribute sets receive different privileges	Ciphertext-bound Boolean policy	Policy enforcement is cryptographic rather than server-only
Integrity	Unauthorized modification is detected	Hash/authenticator or public audit proof	Altered ciphertext must fail verification
Revocation	Former user cannot access newly protected data	Attribute revocation + DEK rotation for new versions	Forward access can be blocked; backward protection needs re-keying
Privacy during audit	Auditor does not require plaintext	Verification metadata / privacy-preserving audit	Integrity can be checked without exposing file content
Efficiency	Large files avoid repeated public-key encryption	Hybrid AES + CP-ABE key encapsulation	ABE cost depends mainly on policy, not file size

The main trade-off is between policy expressiveness and computation. A larger CP-ABE policy increases public-key processing, so the framework should avoid redundant attributes and should consider outsourced ABE techniques where client resources are limited, as demonstrated by Zhang et al. (2017). Revocation is the second major cost. Removing a user from future access is straightforward when fresh keys and policies are used for new versions, but

preventing access to data that the user legitimately obtained earlier is not possible by cryptography alone. Strong backward-security requirements therefore require key rotation, re-encryption, or application-level control over retained copies.

The evaluation also shows why integrity auditing is necessary in addition to encryption. Encryption may keep a corrupted object

confidential, but it does not prove that the cloud still possesses the correct data. Public-auditing work by Wang et al. (2010, 2011) and shared-data auditing by Wang et al. (2014) support the separation of confidentiality and integrity functions used here. The resulting design is modular: an implementation can improve the ABE scheme, auditing method, or revocation mechanism without changing the overall data-sharing workflow.

Conclusion and Discussion

This paper presented a step-by-step methodology for secure data sharing in cloud computing environments using a hybrid combination of symmetric encryption, CP-ABE, integrity verification, revocation, and audit logging. The method addresses the central problem of sharing outsourced information without requiring the cloud provider to be trusted with plaintext or to be the sole enforcer of access policy. ABE supplies fine-grained cryptographic authorization, while symmetric encryption keeps large-file processing practical. Integrity metadata or public auditing provides evidence that outsourced ciphertext remains intact, and explicit revocation handles the changing membership typical of collaborative cloud systems.

The framework is intentionally modular and can be implemented with different ABE or audit constructions. Its principal limitations are the computational cost of complex ABE policies, the management burden of attribute issuance and revocation, and the fact that cryptography cannot force a user to erase plaintext that was legitimately obtained before revocation. Future experimental work should benchmark multiple policy sizes, quantify the benefit of outsourced ABE on constrained clients, and compare direct re-encryption with lazy re-keying strategies. Within the literature boundary of 2017, the proposed methodology provides a practical and defensible foundation for evaluating secure cloud data-sharing systems.

References

- Goyal, V., Pandey, O., Sahai, A., & Waters, B. (2006). Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data. Proceedings of the 13th ACM Conference on Computer and Communications Security. DOI: 10.1145/1180405.1180418.
- Bethencourt, J., Sahai, A., & Waters, B. (2007). Ciphertext-Policy Attribute-Based Encryption. 2007 IEEE Symposium on Security and Privacy, 321-334. DOI: 10.1109/SP.2007.11.
- Ateniese, G., Burns, R., Curtmola, R., Herring, J., Kissner, L., Peterson, Z., & Song, D. (2007). Provable Data Possession at Untrusted Stores. Proceedings of the 14th ACM Conference on Computer and Communications Security. DOI: 10.1145/1315245.1315318.
- Yu, S., Wang, C., Ren, K., & Lou, W. (2010). Achieving Secure, Scalable, and Fine-Grained Data Access Control in Cloud Computing. IEEE INFOCOM 2010. DOI: 10.1109/INFCOM.2010.5462174.
- Wang, C., Wang, Q., Ren, K., & Lou, W. (2010). Privacy-Preserving Public Auditing for Data Storage Security in Cloud Computing. IEEE INFOCOM 2010. DOI: 10.1109/INFCOM.2010.5462173.
- Wang, Q., Wang, C., Ren, K., Lou, W., & Li, J. (2011). Enabling Public Auditability and Data Dynamics for Storage Security in Cloud Computing. IEEE Transactions on Parallel and Distributed Systems, 22(5), 847-859. DOI: 10.1109/TPDS.2010.183.
- Li, M., Yu, S., Zheng, Y., Ren, K., & Lou, W. (2013). Scalable and Secure Sharing of Personal Health Records in Cloud Computing Using Attribute-Based Encryption. IEEE Transactions on Parallel and Distributed Systems, 24(1), 131-143. DOI: 10.1109/TPDS.2012.97.
- Yang, K., Jia, X., Ren, K., & Zhang, B. (2013). DAC-MACS: Effective Data Access Control for Multi-Authority Cloud Storage Systems. IEEE INFOCOM 2013, 2895-2903. DOI: 10.1109/INFCOM.2013.6567100.
- Ruj, S., Stojmenovic, M., & Nayak, A. (2014). Decentralized Access Control with Anonymous Authentication of Data Stored in Clouds. IEEE Transactions on Parallel and Distributed Systems, 25(2), 384-394. DOI: 10.1109/TPDS.2013.38.
- Wang, B., Li, B., & Li, H. (2014). Oruta: Privacy-Preserving Public Auditing for Shared Data in the Cloud. IEEE Transactions on Cloud Computing, 2(1), 43-56. DOI: 10.1109/TCC.2014.2299807.
- Liu, L., Lai, J., Deng, R. H., & Li, Y. (2016). Ciphertext-Policy Attribute-Based Encryption with Partially Hidden Access Structure and Its Application to Privacy-Preserving Electronic Medical Record System in Cloud Environment.

Security and Communication Networks, 9, 4897-4913. DOI: 10.1002/sec.1663.

Zhang, R., Ma, H., & Lu, Y. (2017). Fine-Grained Access Control System Based on Fully

Outsourced Attribute-Based Encryption. Journal of Systems and Software, 125, 344-353. DOI: 10.1016/j.jss.2016.12.018.