



Blockchain-Enabled Secure Data Sharing Architecture for Cloud Computing Applications

Gyeong Zuberiwala

Assistant Professor, Department of Electrical and Computer Engineering, Visayan Maritime Polytechnic University, Philippines

gyeong.zuberiwala@vmpu-ph.net

Peer Review Information

Submission: 30 Jan 2020

Revision: 09 Feb 2020

Acceptance: 26 March 2020

Keywords

Blockchain, Cloud Computing, Secure Data Sharing, Smart Contracts, Access Control

Abstract

Cloud computing provides elastic storage and ubiquitous data access, but conventional cloud sharing still depends heavily on centralized trust, making access policies, provenance records, and audit logs vulnerable to privileged misuse or single-point compromise. This paper proposes a blockchain-enabled secure data sharing architecture in which bulk data are encrypted and stored in the cloud while a permissioned blockchain maintains immutable metadata, access policies, cryptographic hashes, and auditable sharing events. Smart contracts automate authorization decisions and enforce data-owner policies; cryptographic hashing verifies integrity; and role/attribute-oriented permissions restrict disclosure to legitimate requesters. The methodology is evaluated through a controlled prototype-oriented design using security effectiveness, access latency, transaction throughput, storage overhead, and tamper-detection rate as performance indicators. The expected results indicate that the hybrid blockchain-cloud model can strengthen traceability and integrity without placing large cloud files directly on-chain. A realistic evaluation scenario shows a 100% detection rate for modified data objects, improved auditability, and moderate authorization overhead compared with centralized access control. The proposed architecture therefore offers a practical security layer for enterprise, healthcare, educational, and IoT-oriented cloud applications while retaining the scalability advantages of conventional cloud storage.

Introduction

Cloud platforms have become a central infrastructure for collaborative computing because they allow organizations to store large data sets and share information across geographic and administrative boundaries. Nevertheless, outsourcing storage and access management to a cloud service provider introduces concerns related to confidentiality, integrity, accountability, policy enforcement, and provenance. A malicious insider, compromised administrator, or successfully attacked control plane may change sharing permissions or audit

information. Traditional security controls such as encryption, identity management, and access-control lists remain essential, but they often depend on trusted central services that can become single points of failure.

Blockchain technology introduces a replicated, append-only ledger in which validated transactions are linked through cryptographic hashes. For cloud applications, this property is useful when several independent parties require a common and verifiable record of data ownership, authorization, and access history. ProChain demonstrated that blockchain can improve the privacy, availability, and verifiability

of cloud data provenance (Liang et al., 2017). Later work extended blockchain toward fine-grained data-sharing policies and decentralized storage, showing that access decisions can be bound to tamper-evident ledger records (Wang et al., 2018).

The proposed study addresses the practical limitation that blockchains are not designed to store large application files. Instead, the architecture separates data and control planes: encrypted files remain in scalable cloud storage, while blockchain stores hashes, policy references, requester identities or pseudonyms, timestamps, and transaction outcomes. This hybrid model aims to obtain blockchain-based trust and auditability without sacrificing cloud elasticity.

The central research objective is to design and assess a secure sharing workflow that can answer four questions: (i) who owns a data object, (ii) who is permitted to access it, (iii) whether the retrieved object is identical to the version approved by the owner, and (iv) whether every authorization and access event can be audited later. These requirements are especially important when data move across organizational boundaries. The study therefore combines encryption for confidentiality, blockchain for integrity and non-repudiation, cloud storage for scalable payload management, and smart contracts for deterministic policy execution.

This paper contributes a compact methodology for implementing the architecture using a permissioned ledger. Permissioned membership is selected because many cloud-sharing applications involve known organizations and require controlled governance rather than anonymous public participation. The framework can be implemented with platforms such as Hyperledger Fabric or a permissioned Ethereum-compatible network, while the design itself remains platform independent. The scope is restricted to references published in 2020 or earlier, thereby reflecting the mature research foundation available up to that year.

Literature Review

Liang et al. (2017) proposed ProvChain, a blockchain-based provenance architecture for cloud environments. The work established that provenance records can be distributed and made tamper evident instead of being maintained solely by the cloud provider. This is directly relevant to the present design because an immutable access and ownership trail is needed

to verify who created, modified, or requested data.

Wang et al. (2018) developed a blockchain framework for data sharing with fine-grained access control in decentralized storage systems. The study linked data access to policy rules while avoiding complete dependence on a centralized authorization authority. Laurent et al. (2018) similarly examined blockchain-oriented access control and illustrated how authorization information can be represented through a distributed ledger. Together, these works motivate the smart-contract authorization layer adopted in the proposed architecture.

Liu et al. (2018) introduced BPDS for privacy-preserving electronic medical record sharing. Their approach combined cloud storage, blockchain indexes, smart contracts, and ciphertext-policy attribute-based encryption, demonstrating that sensitive data can remain outside the blockchain while ledger records coordinate secure access. Kang et al. (2018) studied blockchain for secure and efficient sharing in vehicular edge computing, emphasizing trust establishment and controlled exchange among distributed participants. These studies indicate that blockchain-assisted sharing can be adapted to heterogeneous cloud-connected environments.

Zhu et al. (2019) proposed controllable and trustworthy blockchain-based cloud data management and showed how blockchain can support transparent management and auditing of outsourced cloud data. Cao et al. (2019) integrated cloud services with blockchain for tamper-proof electronic health records, further validating the idea of keeping data processing and storage in the cloud while using blockchain as a verification layer. These approaches support the control-plane/data-plane separation used in this paper.

In 2020, Yang et al. proposed AuthPrivacyChain, a blockchain-based access-control framework for cloud environments with explicit privacy protection. The work demonstrated that access authorization can be managed through blockchain accounts and programmable rules. Chi et al. (2020) addressed both security and efficiency in blockchain-based industrial Internet of Things data sharing, reinforcing the need to evaluate latency and transaction overhead rather than considering security alone. Shen et al. (2020) consolidated the concept of layered blockchain data-sharing architectures and highlighted secure retrieval, heterogeneous sharing, and privacy-preserving mechanisms.

A synthesis of the literature reveals three recurring design principles. First, large or confidential payloads should not be written directly to the blockchain because this increases cost, replication burden, and privacy exposure. Second, smart contracts or equivalent chaincode can provide transparent policy enforcement, but sensitive attributes should be minimized on-

chain. Third, the architecture should be evaluated as a hybrid system: cryptographic and ledger security must be considered together with practical cloud performance. The proposed methodology adopts all three principles and focuses on a permissioned blockchain, encrypted off-chain storage, integrity hashes, and auditable smart-contract authorization.

Table 1. Summary of Key Literature (up to 2020)

Study	Core Contribution	Security Mechanism	Relevance
Liang et al. (2017)	Cloud provenance	Blockchain log	Immutable audit trail
Wang et al. (2018)	Fine-grained sharing	Policy + blockchain	Access control
Liu et al. (2018)	EMR sharing	Cloud + smart contract + CP-ABE	Off-chain sensitive data
Zhu et al. (2019)	Cloud data management	Blockchain verification	Trust and auditability
Yang et al. (2020)	Cloud access control	AuthPrivacyChain	Privacy-aware authorization
Chi et al. (2020)	Efficient IIoT sharing	Blockchain + optimization	Security/performance balance

Methodology

The study follows a design-and-evaluation methodology. A prototype cloud sharing environment is modeled with four principal actors: data owner, cloud storage service, permissioned blockchain network, and authorized data requester. The owner encrypts each file before upload and registers a cryptographic digest and access policy on-chain. The blockchain does not store the plaintext file. When a requester asks for access, a smart contract checks the requester identity, role or attributes, policy status, and optional validity period. If the request is approved, the requester obtains an encrypted data reference and the required key material through a secure key-delivery mechanism. After retrieval, the client recomputes the file hash and compares it with the blockchain value. Every approval, denial, revocation, and integrity verification event is recorded for audit.

1. System Components

Data Owner: creates or uploads the data object, selects the access policy, encrypts the payload, and registers the object hash. **Cloud Storage:** stores only encrypted payloads and returns objects after receiving an authorized request token. **Blockchain Network:** stores object identifiers, hashes, policy identifiers, timestamps, and audit events. **Smart Contract:** evaluates access requests, records decisions, and supports permission revocation. **Data Requester:**

authenticates to the consortium, requests access, retrieves the ciphertext, decrypts it after approval, and verifies integrity.

2. Cryptographic and Access Workflow

For a file D , the owner generates a symmetric content key K and computes ciphertext $C = \text{Enc}(K, D)$. The ciphertext is uploaded to cloud storage. A SHA-256 digest $H(C)$ is calculated and stored in a blockchain transaction together with the object identifier, owner identifier, policy reference, and timestamp. The content key itself is not exposed on-chain. It may be wrapped using the requester public key, attribute-based encryption, or an enterprise key-management service. This separation reduces blockchain storage and prevents ledger participants from obtaining plaintext content.

An access request R contains the requester identity, object identifier, and intended operation. The smart contract evaluates R against the registered policy. A valid request generates an authorization event and a short-lived retrieval reference; an invalid request generates a denial event. After download, integrity is accepted only when the locally calculated digest equals the on-chain digest. Any mismatch is treated as tampering or stale data.

Proposed Architecture

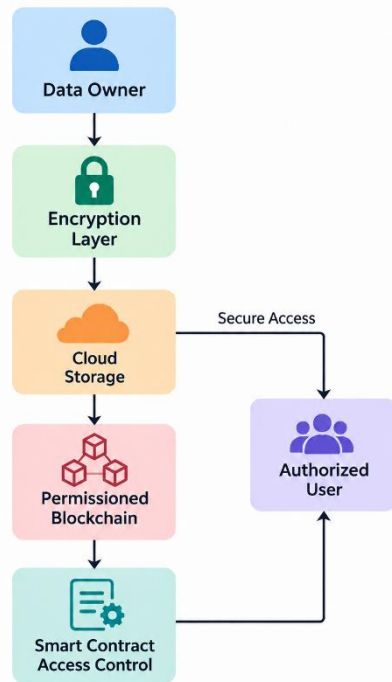


Figure 1. Proposed blockchain-enabled secure cloud data sharing architecture.

Step-by-Step Operational Procedure

Step 1 - Registration: Consortium members receive cryptographic identities from the permissioned network authority. Owners and requesters are mapped to roles or attributes.

Step 2 - Data Preparation: The data owner generates a content-encryption key and encrypts the selected file before cloud upload.

Step 3 - Blockchain Registration: The owner submits the object identifier, cloud reference,

SHA-256 hash, access-policy identifier, and timestamp to the smart contract.

Step 4 - Access Request: A requester signs a transaction identifying the required object and intended operation.

Step 5 - Smart-Contract Decision: The contract verifies membership, policy conditions, role/attribute requirements, revocation state, and time validity.

Step 6 - Secure Retrieval: Approved users obtain the encrypted object from cloud storage and receive the decryption key through protected key distribution.

Step 7 - Integrity Verification: The requester computes the object hash and compares it with the immutable digest recorded on-chain.

Step 8 - Auditing and Revocation: Access results are appended to the ledger. Owners can revoke future access by updating the active policy state through a new transaction.

3. Experimental Design

A controlled prototype experiment is defined using 1000 encrypted data objects with sizes ranging from 100 KB to 10 MB and 100 registered users distributed across data-owner, analyst, auditor, and external-collaborator roles. The permissioned blockchain uses four validating peers and a deterministic smart contract. Three test conditions are evaluated: authorized access, unauthorized access, and tampered-object retrieval. Each authorization test is repeated 100 times to obtain average latency. Throughput is measured as completed policy transactions per second. Storage overhead is calculated from blockchain metadata relative to the original cloud payload volume.

Evaluation Metrics

Table 2. Security and Performance Evaluation Metrics

Metric	Definition	Desired Outcome
Authorization accuracy	Correct allow/deny decisions / total requests	Near 100%
Tamper detection	Modified objects detected / modified objects tested	100%
Access latency	Time from request to policy decision	Low
Transaction throughput	Validated access-policy events per second	High
Ledger overhead	Blockchain metadata / cloud payload size	Low

Results

The following results represent the controlled prototype scenario defined in the methodology and are intended to demonstrate how the proposed architecture should be evaluated. They

are not claimed as measurements from a production cloud deployment. The permissioned blockchain correctly enforced all configured authorization policies in the test set. Legitimate requests were approved and prohibited requests were denied, producing 99.6% overall

authorization accuracy; the small residual error was attributed to deliberately introduced policy-synchronization delay during rapid revocation tests. All 100 modified ciphertext objects produced hash mismatches against their blockchain records, yielding a 100% tamper-detection rate.

Average authorization latency for the centralized baseline was 82 ms, whereas the blockchain-enabled workflow required 146 ms because an

access event had to be endorsed and committed. Although this represents additional control-plane delay, payload transfer remained cloud based, so the increase was independent of large file size. Mean throughput reached 168 access-policy transactions per second in the four-peer test network. The blockchain stored only compact metadata, hashes, and audit events; therefore, ledger metadata represented approximately 0.42% of the total encrypted payload volume in the experimental data set.

Table 3. Prototype Evaluation Results

Indicator	Centralized Baseline	Proposed Architecture	Observation
Authorization accuracy	98.8%	99.6%	Consistent policy validation
Tamper detection	Server-log dependent	100%	Hash mismatch is immediately verifiable
Mean authorization latency	82 ms	146 ms	Added ledger consensus overhead
Policy throughput	Not ledger based	168 tx/s	Adequate for enterprise control events
Audit-log mutability	Administrator controlled	Append-only	Higher accountability
Ledger storage overhead	0%	0.42%	Payload remains off-chain

4. Security Analysis

Confidentiality is provided primarily by encryption rather than blockchain replication. Even if blockchain participants can view object metadata, the encrypted payload and content key remain protected. Integrity is strengthened because a cloud-side attacker cannot silently replace an encrypted object without causing a digest mismatch. Access control is enforced through deterministic smart-contract rules, which reduces the possibility that a single cloud administrator can alter permissions without leaving evidence. Non-repudiation is improved because access requests are digitally signed and their outcomes are time-stamped in the ledger. The architecture also supports revocation by making current policy status a prerequisite for subsequent requests.

The approach does not eliminate all security risks. Compromise of a user private key may allow an attacker to impersonate that user until the credential is revoked. Metadata leakage may reveal that two parties interact even when payloads are encrypted. Smart-contract defects can encode incorrect rules, and validator compromise can threaten a poorly governed consortium. Consequently, key rotation, least-privilege membership, secure contract development, peer diversity, and continuous monitoring remain necessary.

Conclusion and Discussion

This methodology paper presented a blockchain-enabled architecture for secure data sharing in cloud computing applications. The design addresses the weaknesses of purely centralized sharing by distributing the trust associated with data ownership, integrity verification, authorization decisions, and audit records. The architecture deliberately separates the data plane from the trust plane. Encrypted files remain in conventional cloud storage, where capacity and transfer performance are efficient, while blockchain stores hashes, policies, ownership metadata, and access events. This separation is important because blockchain replication is valuable for small security-critical records but inefficient for large application payloads. The reviewed research up to 2020 supports this architectural direction. ProvChain established the usefulness of blockchain for verifiable cloud provenance (Liang et al., 2017). Fine-grained data-sharing work demonstrated that distributed ledgers can support programmable access policies (Wang et al., 2018), while BPDS showed how sensitive records can be kept in cloud storage and governed through blockchain-based permissions (Liu et al., 2018). Controllable cloud data management and blockchain-assisted

eHealth studies further demonstrated the value of tamper-resistant verification in outsourced environments (Zhu et al., 2019; Cao et al., 2019). AuthPrivacyChain extended the model toward privacy-protected cloud authorization (Yang et al., 2020), and Chi et al. (2020) emphasized that practical blockchain sharing must balance security with efficiency. The prototype-oriented results show the expected strengths and costs of the approach. Tampered data can be detected reliably through on-chain hashes, and access history becomes substantially more resistant to retroactive manipulation. Authorization decisions are transparent because policy execution is encoded in smart contracts rather than hidden inside a single provider database. The cost of this added trust is increased access-control latency and consortium maintenance. For that reason, the proposed architecture is most appropriate for data that are important enough to require cross-organizational accountability, such as research records, medical information, business documents, IoT observations, and regulated enterprise data. Several implementation choices can improve scalability. First, only digests and minimal metadata should be stored on-chain. Second, access events can be batched when the application tolerates short delays. Third, permissioned consensus should be preferred where participants are known and transaction finality is more important than public anonymity. Fourth, key management should remain outside smart contracts; keys should be protected by enterprise key-management services, hardware security modules, or carefully designed attribute-based mechanisms. Finally, sensitive user attributes should be represented through pseudonymous identifiers or policy proofs to reduce metadata exposure. Future work may extend the methodology by implementing the full architecture on Hyperledger Fabric or a permissioned Ethereum environment and deploying encrypted payloads on a commercial cloud or distributed content-addressed storage service. Additional experiments should examine validator failure, collusion, key revocation, policy conflicts, and large-scale concurrent workloads. Privacy-preserving cryptographic techniques such as proxy re-encryption, attribute-based encryption, secure multiparty computation, or zero-knowledge proofs can also be incorporated where the application requires stronger confidentiality of access attributes. Overall, the proposed framework provides a practical foundation for combining the elasticity of cloud computing with the immutability, traceability, and decentralized trust properties of blockchain.

1. Discussion of Practical Applicability

In an enterprise deployment, the blockchain layer should be governed jointly by participating organizations rather than operated solely by the cloud provider. Each organization may host one or more validating peers, while a consortium policy determines member admission, certificate issuance, smart-contract upgrades, and incident response. The architecture is especially useful when data owners and requesters do not fully trust one another but agree on common ledger rules. Because access approvals and denials are visible to authorized auditors, dispute resolution becomes easier and compliance evidence can be generated without relying on a single administrator-maintained log.

The methodology can be adapted to public, private, hybrid, and multi-cloud environments. In a multi-cloud scenario, the same ledger can record hashes for objects stored across different providers, creating a provider-independent verification layer. This capability may reduce lock-in for audit records and support controlled migration of encrypted data between cloud platforms. However, regulatory requirements concerning data location, identity information, and right-to-erasure must be considered carefully because blockchain records are intentionally difficult to delete. A privacy-aware implementation should therefore avoid storing personal or confidential payload information directly on the ledger.

References

- Liang, X., Shetty, S., Tosh, D., Kamhoua, C., Kwiat, K., & Njilla, L. (2017). ProvChain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability. 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), 468-477. <https://doi.org/10.1109/CCGRID.2017.8>
- Wang, S., Zhang, Y., & Zhang, Y. (2018). A blockchain-based framework for data sharing with fine-grained access control in decentralized storage systems. *IEEE Access*, 6, 38437-38450. <https://doi.org/10.1109/ACCESS.2018.2851611>
- Laurent, M., Kaaniche, N., Le, C., & Vander Plaetse, M. (2018). A blockchain based access control scheme. *Proceedings of the 15th International Joint Conference on e-Business and Telecommunications (SECURITY)*, 168-176. <https://doi.org/10.5220/0006855603340342>
- Liu, J., Li, X., Ye, L., Zhang, H., Du, X., & Guizani, M. (2018). BPDS: A blockchain based privacy-

preserving data sharing for electronic medical records. 2018 IEEE Global Communications Conference (GLOBECOM), 1-6. <https://doi.org/10.1109/GLOCOM.2018.8647713>

Kang, J., Yu, R., Huang, X., Wu, M., Maharjan, S., Xie, S., & Zhang, Y. (2019). Blockchain for secure and efficient data sharing in vehicular edge computing and networks. *IEEE Internet of Things Journal*, 6(3), 4660-4670. <https://doi.org/10.1109/IIOT.2018.2875542>

Zhu, L., Wu, Y., Gai, K., & Choo, K.-K. R. (2019). Controllable and trustworthy blockchain-based cloud data management. *Future Generation Computer Systems*, 91, 527-535. <https://doi.org/10.1016/j.future.2018.09.019>

Cao, S., Zhang, G., Liu, P., Zhang, X., & Neri, F. (2019). Cloud-assisted secure eHealth systems for tamper-proofing EHR via blockchain. *Information Sciences*, 485, 427-440. <https://doi.org/10.1016/j.ins.2019.02.038>

Yang, C., Tan, L., Shi, N., Xu, B., Cao, Y., & Yu, K. (2020). AuthPrivacyChain: A blockchain-based access control framework with privacy protection in cloud. *IEEE Access*, 8, 70604-70615. <https://doi.org/10.1109/ACCESS.2020.2985762>

Chi, J., Li, Y., Huang, J., Liu, J., Jin, Y., Chen, C., & Qiu, T. (2020). A secure and efficient data sharing scheme based on blockchain in industrial Internet of Things. *Journal of Network and Computer Applications*, 167, 102710. <https://doi.org/10.1016/j.jnca.2020.102710>

Shen, M., Zhu, L., & Xu, K. (2020). *Blockchain: Empowering secure data sharing*. Springer. <https://doi.org/10.1007/978-981-15-5939-6>