INTERNATIONAL JOURNAL OF EXPLORING EMERGING TRENDS IN ENGINEERING
Peer-Reviewed, Refereed, Indexed and International Journal, https://ijoeete.com/
(Peer Review Journal as per UGC 2025 Standard Guidelines)
|ISSN No. 2394-0573 |Volume: 12, Issue: 4 | October – December 2025

BLOCKCHAIN-ENABLED SECURE AND EFFICIENT DATA EXCHANGE ACROSS DECENTRALIZED STORAGE SYSTEMS

[1]Brijendra Yadav, [2]Dr. Rohini Mudliyar (Assistant Professor)
[1]Research Scholar, [2]Research Supervisor
[1-2]Department of Computer Science, Bharti University, Durg (C.G)

Abstract
Decentralized storage networks such as IPFS, Swarm, Storj and Arweave remove the single point of control found in cloud storage, but they do not by themselves provide access control, accountability or a way to exchange data between networks that use different addressing and incentive schemes. Blockchains are widely proposed as the missing coordination layer. This paper analyses how far a blockchain can make data exchange across decentralized storage systems both secure and efficient, and where the limits lie. A four-layer reference architecture is defined, in which encrypted data stay off-chain and only a content digest, an access policy and an audit trail are placed on the ledger. The architecture is evaluated analytically using protocol constants and published measurements. Storing 1 MB of data directly in Ethereum contract storage would need about 724 million gas, equal to the full capacity of twenty-four blocks, whereas a three-slot registry record needs about 87,300 gas, a reduction of roughly four orders of magnitude. An integrity proof for one 256 kB chunk of a 1 TB file is only 704 bytes. A permissioned ledger with a reported throughput of more than 3,500 transactions per second can register about 300 million exchange records a day, against about 2.5 million for a public chain used at full capacity. On the storage side, measurements of the IPFS network show that 45.5 per cent of peers are never reachable and only 1.4 per cent are reliably online, so availability, not confidentiality, is the weakest property. A probabilistic model shows that when nodes are online only half the time, three-fold replication gives 87.5 per cent file availability and a 10-of-20 erasure code gives only 58.8 per cent, while a 29-of-80 code gives 99.5 per cent at lower storage overhead than replication. The paper concludes that the blockchain should carry control information only, that the choice of redundancy scheme matters more for efficiency than the choice of ledger, and that key management and cross-network addressing remain the principal open problems.

Keywords: blockchain; decentralized storage; IPFS; smart contracts; proxy re-encryption; erasure coding; content addressing; data exchange; interoperability

1. Introduction

The volume of data exchanged between organizations has grown faster than the trust between them. Hospitals share records with insurers and laboratories, manufacturers share design files with suppliers, and research groups share datasets across institutions. In the conventional model this exchange passes through a cloud provider who stores the data, enforces access rules and keeps the logs. The provider is therefore a single point of failure, a single point of attack and a party that every participant has to trust.

Decentralized storage networks were designed to remove that dependence. In the InterPlanetary File System (IPFS), a file is split into blocks, each block is named by the hash of its content, and any peer holding a block can serve it (Benet, 2014). Swarm, Storj, Filecoin and Arweave follow related designs with different approaches to incentives and persistence (Daniel and Tschorsch, 2022). Content addressing gives these networks a strong integrity property: data that do not match the requested hash are rejected by the receiver.

INTERNATIONAL JOURNAL OF EXPLORING EMERGING TRENDS IN ENGINEERING
Peer-Reviewed, Refereed, Indexed and International Journal, https://ijoeete.com/
(Peer Review Journal as per UGC 2025 Standard Guidelines)
|ISSN No. 2394-0573 |Volume: 12, Issue: 4 | October – December 2025

Integrity, however, is only one of the requirements of data exchange. A content-addressed network does not say who is allowed to read a file, does not record who has read it, does not guarantee that anyone continues to store it, and does not help when the sender uses one network and the receiver another. Blockchains have been proposed to fill these gaps. A blockchain offers an append-only record that no single participant controls, and smart contracts allow access rules to be enforced by code that all parties can inspect (Nakamoto, 2008; Wood, 2014). A number of schemes combine a blockchain with IPFS and with cryptographic access control, for example attribute-based encryption (Wang et al., 2018) or secret sharing (Naz et al., 2019).

These schemes are usually presented one at a time, each with its own prototype, and their results are difficult to compare. Two general questions are left open. First, what should be placed on the chain and what must stay off it, and what does each choice cost? Second, when data are exchanged across several storage networks, which component limits security and which limits efficiency? This paper addresses both questions analytically. Its objectives are:

- to define a reference architecture for blockchain-enabled data exchange across heterogeneous decentralized storage systems;
- to quantify the cost of on-chain and off-chain data placement from protocol constants;
- to compare public and permissioned ledgers on the throughput available for exchange records;
- to analyse the availability of data under replication and erasure coding when storage nodes are unreliable;
- to assess the security of the architecture against a defined set of threats; and
- to identify the open problems that limit secure and efficient exchange.

2. Background and Related Work

2.1 Decentralized storage systems

Decentralized storage systems share three building blocks: content addressing, a peer-to-peer overlay for locating content, and a redundancy mechanism. They differ in how each is realised. Table 1 summarises four representative systems on the basis of a published comparative survey.

Table 1. Characteristics of representative decentralized storage systems

System	Unit and addressing	Lookup	Redundancy	Persistence	Incentive
IPFS	Blocks in a Merkle DAG; content identifier	Kademlia DHT and opportunistic exchange	Passive replication and caching	Not guaranteed	Optional, through Filecoin
Swarm	Chunks in a Merkle tree; chunk hash	Kademlia DHT	Replication and erasure codes	Not guaranteed	Ethereum-based payments
Storj	Encrypted segments split into pieces	Central satellite nodes	Reed-Solomon erasure codes	Fixed, paid lifetime	Token payment for storage time
Arweave	Files stored in a blockweave	Opportunistic	Replication	Intended to be permanent	One-time payment

Two points in Table 1 are important for exchange between systems. The systems name content differently, so the identifier of a file in one network is meaningless in another. And only some of them offer any guarantee that the data will still be stored tomorrow. IPFS, the most widely used, leaves persistence to

whoever chooses to keep ("pin") the data.

The largest published measurement of IPFS observed 198,964 peers in 152 countries and 2,715 autonomous systems (Trautwein et al., 2022). Three quarters of retrievals from Europe completed in under two seconds. The same study found that 45.5 per cent of observed peers were never reachable, that only 1.4 per cent could be classed as reliable with more than 90 per cent uptime, and that the ten largest autonomous systems held 64.9 per cent of the IP addresses. The network is fast enough for practical use but far less stable and less evenly distributed than the word "decentralized" suggests.

2.2 Blockchains and smart contracts

A blockchain is a replicated, append-only ledger whose state is agreed by a consensus protocol. Public blockchains such as Bitcoin and Ethereum admit any participant and pay for security with limited throughput: Bitcoin produces a block about every ten minutes and Ethereum one every twelve seconds. Permissioned blockchains restrict participation to known organizations and can use faster consensus. Hyperledger Fabric reported an end-to-end throughput of more than 3,500 transactions per second with an average latency of about 500 ms on five peers in a single data centre (Androulaki et al., 2018). Smart contracts are programs stored on the ledger and executed by every validating node, which makes them suitable for rules that all parties must be able to verify.

2.3 Cryptographic access control

Because storage nodes are untrusted, data must be encrypted before upload, and the problem of access control becomes a problem of key distribution. Three approaches recur in the literature. In attribute-based encryption the data key is encrypted under a policy over attributes, and only users whose attributes satisfy the policy can recover it (Wang et al., 2018). In secret sharing the key or the content hash is split into shares so that a threshold number are needed to reconstruct it (Naz et al., 2019). In proxy re-encryption a semi-trusted proxy transforms a ciphertext for the owner into a ciphertext for the requester without learning the plaintext (Blaze et al., 1998; Ateniese et al., 2006). Proxy re-encryption fits the exchange setting well, since the owner can grant access to a new requester without re-encrypting or re-uploading the file.

2.4 Research gap

Existing schemes demonstrate that the combination of a blockchain with one storage network works. They rarely consider exchange across several storage networks, they seldom separate the cost of the ledger from the cost of storage, and they generally assume that stored data remain available. The analysis that follows addresses these three points.

3. Reference Architecture and Method

3.1 Architecture

The reference architecture has four layers, shown in Figure 1.

- **Storage layer.** One or more decentralized storage networks hold the encrypted data. Nothing in this layer is trusted for confidentiality.
- **Cryptographic layer.** The owner encrypts each file with a fresh symmetric key using an authenticated cipher such as AES-256-GCM. The symmetric key is protected by proxy re-encryption or an equivalent scheme.
- **Ledger layer.** Smart contracts maintain a content registry, an access-control list and an audit log. The registry maps a storage-independent digest of the ciphertext to its owner and to its locations in each storage network.
- **Exchange layer.** A protocol between owner and requester that uses the three lower layers.

INTERNATIONAL JOURNAL OF EXPLORING EMERGING TRENDS IN ENGINEERING
Peer-Reviewed, Refereed, Indexed and International Journal, https://ijoeete.com/
(Peer Review Journal as per UGC 2025 Standard Guidelines)
|ISSN No. 2394-0573 |Volume: 12, Issue: 4 | October – December 2025

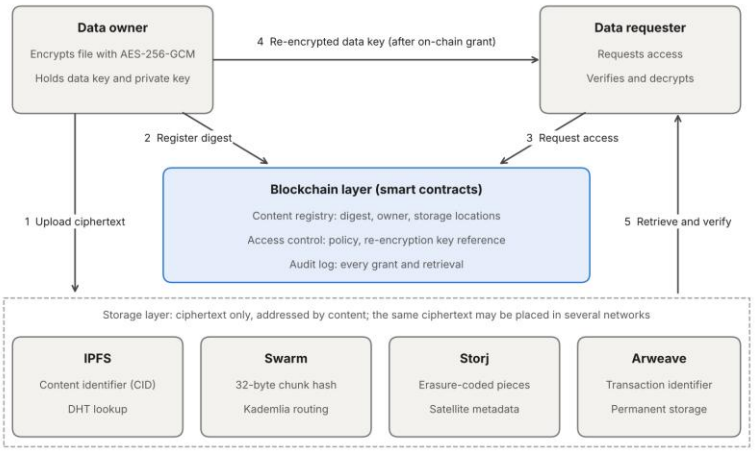


Figure 1. Reference architecture for blockchain-enabled data exchange across decentralized storage systems.

An exchange proceeds in five steps.

1. The owner encrypts the file, uploads the ciphertext to one or more storage networks and obtains a location identifier from each.
2. The owner registers on the ledger the SHA-256 digest of the ciphertext, the location identifiers and the access policy.
3. The requester submits an access request to the access-control contract, which checks it against the policy and records the decision.
4. If access is granted, a re-encryption key is released and the data key is re-encrypted for the requester. The grant is recorded on the ledger.
5. The requester resolves a location from the registry, retrieves the ciphertext from whichever network is reachable, checks it against the registered digest and decrypts it.

The registry is what makes exchange across storage systems possible. The digest of the ciphertext is independent of any network, so the same record can point to a copy in IPFS, a copy in Swarm and a copy in Storj, and the requester can verify whichever copy is retrieved.

3.2 Method of analysis

The architecture was not implemented. It was evaluated analytically, using three kinds of input.

- **Protocol constants.** The Ethereum gas schedule: 20,000 gas to set a 32-byte storage slot from zero to a non-zero value, 2,100 gas for the first access to a slot in a transaction, 16 gas per non-zero byte of transaction data, and 21,000 gas as the base cost of a transaction (Wood, 2014, as amended). A block gas limit of 30 million, the value in force on Ethereum for most of 2021 to 2024, was used as the reference capacity.
- **Published measurements.** Throughput and latency of Hyperledger Fabric (Androulaki et al., 2018), measurements of the IPFS network (Trautwein et al., 2022), and smart contract costs of a prototype (Naz et al., 2019).
- **A probabilistic availability model.** Each storage node is assumed to be online independently with probability a. A file stored as n pieces of which any k suffice is available with probability:

$$A(k, n, a) = \Sigma_{i=k\ldots n}\, C(n, i) \cdot a^{i} \cdot (1 - a)^{n-i}$$

Replication with r copies is the special case $k = 1$, $n = r$. The storage overhead of a scheme is n / k. The size of a Merkle integrity proof for one chunk of a file of N chunks is $\lceil \log_2 N \rceil$ hash values of 32 bytes each.

INTERNATIONAL JOURNAL OF EXPLORING EMERGING TRENDS IN ENGINEERING
Peer-Reviewed, Refereed, Indexed and International Journal, https://ijoeete.com/
(Peer Review Journal as per UGC 2025 Standard Guidelines)
|ISSN No. 2394-0573 |Volume: 12, Issue: 4 | October – December 2025

4. Results and Discussion

4.1 Cost of placing data on the chain

Table 2 gives the gas needed to place data of increasing size on Ethereum, either in contract storage or as transaction data, and the share of a 30-million-gas block that this represents.

Table 2. Gas required to place data on Ethereum, by size and method

Data placed on chain	Contract storage (gas)	Blocks of 30 million gas	Transaction data (gas)	Share of one block (%)
Registry record, 96 bytes	66,300	0.002	1,536	0.005
1 KB	707,200	0.02	16,384	0.05
10 KB	7,072,000	0.24	163,840	0.55
100 KB	70,720,000	2.36	1,638,400	5.5
1 MB	724,172,800	24.1	16,777,216	55.9

Storing one megabyte in contract storage would consume about 724 million gas, the whole capacity of twenty-four consecutive blocks. Even as transaction data, which is cheaper because it does not enter the contract state, one megabyte takes more than half a block. A registry record of three storage slots, holding a 32-byte digest, the owner address with a timestamp, and a pointer to the key material, costs 66,300 gas, or about 87,300 gas with the base transaction cost. The ratio between storing a megabyte and storing its record is about 10,900 to one.

The design rule that follows is firm: the ledger must hold control information only. This also explains a common pattern in published prototypes. The prototype of Naz et al. (2019) needed 1,808,235 gas to deploy its storage contract, a one-time cost equal to about twenty registry records, after which each file added only a small transaction. The cost of a blockchain-based exchange system is dominated by the number of exchange events, not by the volume of data exchanged.

Keeping only a digest on the chain does not weaken integrity. Table 3 gives the size of the proof needed to verify one 256 kB chunk, the default chunk size in IPFS, against the root hash of a file.

Table 3. Size of a Merkle integrity proof for one 256 kB chunk

File size	Number of chunks	Hashes in proof	Proof size (bytes)
1 MB	4	2	64
100 MB	400	9	288
1 GB	4,096	12	384
100 GB	409,600	19	608
1 TB	4,194,304	22	704

The proof grows with the logarithm of the file size. A requester can verify any chunk of a terabyte file with 704 bytes of proof and a single 32-byte root held on the ledger. Integrity verification is therefore not a source of inefficiency at any realistic scale.

4.2 Throughput of the ledger

Each exchange places at least one record on the ledger, and normally three: registration, grant and retrieval receipt. The rate at which records can be written sets an upper limit on the rate of exchange. Table 4 compares three ledgers.

INTERNATIONAL JOURNAL OF EXPLORING EMERGING TRENDS IN ENGINEERING
Peer-Reviewed, Refereed, Indexed and International Journal, https://ijoeete.com/
(Peer Review Journal as per UGC 2025 Standard Guidelines)
|ISSN No. 2394-0573 |Volume: 12, Issue: 4 | October – December 2025

Table 4. Capacity of three ledgers for exchange records

Ledger	Type	Block interval	Throughput (transactions per second)	Records per day (millions)
Bitcoin	Public, proof of work	About 10 minutes	About 7	0.6
Ethereum	Public, proof of stake	12 seconds	About 29 registry records	2.5
Hyperledger Fabric	Permissioned	Configurable	More than 3,500	302

The Ethereum figure is calculated, not measured: a 30-million-gas block holds about 343 registry transactions of 87,300 gas, which is about 29 per second if the whole block were used for this one application. In practice the block is shared with all other users of the chain, so the usable rate is a small fraction of this. The Fabric figure is the reported peak for a simple asset-transfer workload in one data centre, and a deployment across organizations and regions would be slower. Even with these qualifications the difference is two orders of magnitude.

The choice of ledger is a trade-off and not a ranking. A public chain offers openness and resistance to collusion among a known group, at the price of low throughput, a fee for every record and, on Ethereum, about thirteen minutes to economic finality. A permissioned ledger offers high throughput, sub-second confirmation and no per-record fee, but its guarantees are only as strong as the independence of the member organizations. Exchange among a defined consortium, such as a group of hospitals, is naturally served by a permissioned ledger. Open exchange among parties who do not know each other needs a public chain, preferably with records batched or moved to a second-layer network so that many exchanges share one base-layer transaction.

Batching. The limit of a public chain can be raised by committing many records in one transaction. If the owner or an aggregator collects m exchange records, builds a Merkle tree over them and writes only the root to the ledger, one transaction of about 43,100 gas (the base cost plus one storage slot) covers all m records. With $m = 1{,}024$ the cost falls from 87,300 gas to about 42 gas per record, and each record remains individually verifiable with a proof of ten hashes, or 320 bytes, held off-chain. A 30-million-gas block could then carry about 696 batches, or some 713,000 records. The price is a delay while the batch fills and a dependence on the aggregator to publish the proofs, so batching suits audit records better than access grants, which a requester needs promptly.

A worked example. Consider a consortium that performs 50,000 exchanges a day, each producing three ledger records, or 150,000 records a day. Written individually to Ethereum these would need about 13.1 billion gas a day. The chain produces 7,200 blocks a day with a combined capacity of 216 billion gas, so this one application would occupy about 6 per cent of the entire network, which is neither affordable nor reasonable. Batched in groups of 1,024, the same records need 147 transactions and about 6.3 million gas a day, a fifth of one block. On the permissioned ledger the 150,000 records are about 0.05 per cent of the daily capacity without any batching. The example shows that a public chain is practical for exchange records only with batching, and that a permissioned ledger has capacity to spare.

4.3 Availability of the stored data

A valid access grant is worthless if the ciphertext cannot be retrieved. Table 5 gives the availability of a file under five redundancy schemes for three values of node availability, and Figure 2 shows the same model over the whole range.

INTERNATIONAL JOURNAL OF EXPLORING EMERGING TRENDS IN ENGINEERING
Peer-Reviewed, Refereed, Indexed and International Journal, https://ijoeete.com/
(Peer Review Journal as per UGC 2025 Standard Guidelines)
|ISSN No. 2394-0573 |Volume: 12, Issue: 4 | October – December 2025

Table 5. File availability under replication and erasure coding (%)

Scheme	Storage overhead	Nodes online 50% of the time	Nodes online 70% of the time	Nodes online 90% of the time
3 replicas	3.0x	87.5	97.3	99.9
5 replicas	5.0x	96.9	99.76	99.999
Erasure code, 10 of 20	2.0x	58.8	98.3	99.9999
Erasure code, 20 of 40	2.0x	56.3	99.76	More than 99.9999
Erasure code, 29 of 80	2.76x	99.5	More than 99.9999	More than 99.9999

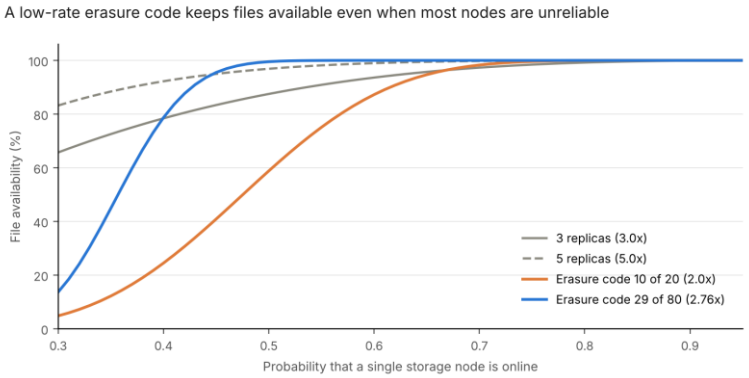


Figure 2. File availability against node availability for four redundancy schemes.

Three results stand out. First, when nodes are reliable (online 90 per cent of the time), erasure coding is far more efficient than replication: a 10-of-20 code gives better availability than five replicas at two-fifths of the storage. This agrees with the durability comparison published by Storj, in which an 8-of-16 code at an expansion factor of two gave 99.995 per cent durability against 98.248 per cent for two replicas at the same expansion (Storj Labs, 2024).

Second, the advantage reverses when nodes are unreliable. An erasure code needs k of n pieces, so if the fraction of nodes online falls to the code rate k / n, availability collapses to about one half. At 50 per cent node availability the 10-of-20 code gives 58.8 per cent, well below the 87.5 per cent of plain three-fold replication. Third, a low-rate code restores the advantage: 29 of 80 gives 99.5 per cent availability at 50 per cent node availability, with a storage overhead of 2.76, lower than that of three replicas.

This matters because open peer-to-peer networks are at the unreliable end of the range. With 45.5 per cent of IPFS peers never reachable and 1.4 per cent reliably online, a file held by a few arbitrary peers cannot be assumed to be retrievable. The model assumes that nodes fail independently, which is optimistic: with the ten largest autonomous systems holding 64.9 per cent of addresses, the failure of one hosting provider would remove many nodes at once, and real availability would be lower than Table 5 suggests.

Redundancy can be supported by proofs of storage. In Filecoin a storage provider must first prove that it has created a unique physical copy of the data (proof of replication) and then prove at regular intervals that it still holds that copy (proof of spacetime), with the proofs checked on the blockchain and a penalty applied when one is missed (Protocol Labs, 2017). Storj uses periodic audits of randomly chosen stripes by its satellite nodes. Such mechanisms turn a hope that the data are stored into an obligation that can be checked, and they are the appropriate choice when the exchanged data must remain available for a fixed period. They do not remove the need for redundancy, since a proof can only report that a copy has been

INTERNATIONAL JOURNAL OF EXPLORING EMERGING TRENDS IN ENGINEERING
Peer-Reviewed, Refereed, Indexed and International Journal, https://ijoeete.com/
(Peer Review Journal as per UGC 2025 Standard Guidelines)
|ISSN No. 2394-0573 |Volume: 12, Issue: 4 | October – December 2025

lost, not restore it.

For exchange across storage systems the implication is practical. Placing copies of the ciphertext in two or three different networks, and recording all the locations in the registry, gives independence of failure that no single network provides. The storage cost of this is modest because the ciphertext is the same in each network and only the location identifiers differ.

4.4 End-to-end efficiency

The time for one exchange is the sum of six components: encryption, upload, ledger confirmation, lookup, download and decryption.

$$T_{exchange} = T_{enc} + T_{upload} + T_{ledger} + T_{lookup} + T_{download} + T_{dec}$$

Symmetric encryption and decryption proceed at hundreds of megabytes per second on ordinary hardware and are negligible for all but very large files. Upload and download depend on bandwidth and file size, as they would with any storage service. The two components specific to this architecture are the ledger confirmation and the lookup in the storage network. On a permissioned ledger confirmation takes about half a second; on Ethereum inclusion in a block takes about twelve seconds and finality about thirteen minutes. Lookup in the IPFS distributed hash table completed within two seconds for three quarters of the European retrievals measured.

For small files, therefore, the ledger dominates the exchange time on a public chain and the storage lookup dominates on a permissioned one. For large files, transfer time dominates in both cases and the overhead of the blockchain becomes insignificant. The architecture is least efficient for many small exchanges on a public chain, which is exactly the case in which batching of records is most useful.

4.5 Security analysis

Table 6 sets out the main threats to a data exchange and the component of the architecture that addresses each.

Table 6. Threats and countermeasures in the reference architecture

Threat	Countermeasure	Residual risk
Storage node reads the data	Client-side authenticated encryption; nodes hold ciphertext only	Metadata such as file size and access pattern remain visible
Storage node alters or substitutes data	Content addressing and digest on the ledger; Merkle proof per chunk	None for integrity; the node can still withhold the data
Storage node discards the data	Redundancy across nodes and across networks; proofs of storage where offered	Persistence is not guaranteed in networks without incentives
Unauthorized party requests access	Policy enforced by smart contract; key released only after an on-chain grant	Errors in contract code; compromise of the owner's private key
Proxy colludes with a requester	Proxy re-encryption reveals no plaintext to the proxy; threshold of several proxies	A granted requester can always copy the plaintext
Party denies having sent or received	Signed, time-stamped records on an append-only ledger	The ledger proves the grant, not that the data were read
Access must be withdrawn	Grant revoked on the ledger; data re-encrypted under a new key	Copies already decrypted cannot be recalled

The analysis shows an asymmetry. Integrity and non-repudiation are strongly provided, because they rest on hash functions and on the ledger. Confidentiality is provided as long as keys are managed correctly, so the security of the whole system reduces to the security of key management. Availability is the weakest

INTERNATIONAL JOURNAL OF EXPLORING EMERGING TRENDS IN ENGINEERING
Peer-Reviewed, Refereed, Indexed and International Journal, https://ijoeete.com/
(Peer Review Journal as per UGC 2025 Standard Guidelines)
|ISSN No. 2394-0573 |Volume: 12, Issue: 4 | October – December 2025

property, since no cryptographic mechanism can force a node to return data. Two further limits deserve mention.

- **Revocation is partial.** Withdrawing a grant prevents future key release, but a requester who has already decrypted the file keeps the plaintext. This is a limit of any system that delivers data to its users and cannot be solved by the ledger.
- **The ledger is public and permanent.** Access policies, addresses and the timing of exchanges are visible to every ledger participant, and can reveal who deals with whom even though the data are encrypted. Records cannot be erased, which conflicts with legal rights to deletion of personal data. Personal data, and identifiers that can be linked to a person, should therefore never be written to the ledger, even in hashed form where the input can be guessed.

4.6 Exchange across heterogeneous storage systems

Three obstacles stand between the present state of the art and true exchange across storage systems.

- **Addressing.** An IPFS content identifier encodes a hash of a Merkle DAG built with a particular chunk size and layout; a Swarm reference is the root of a differently constructed tree; Storj and Arweave use identifiers that are not derived from the content in the same way. The same file therefore has unrelated names in each network. The registry of Section 3.1 resolves this by using the plain SHA-256 digest of the ciphertext as the primary key and treating the network-specific identifiers as locations.
- **Ledgers.** The storage networks are tied to different blockchains or to none. Where the parties to an exchange use different ledgers, a record on one must be made verifiable on the other. The known mechanisms are notary schemes, relays that verify block headers of one chain on another, and hashed time-lock contracts for atomic exchange (Herlihy, 2018; Belchior et al., 2021). Each adds latency and a further trust assumption.
- **Incentives.** A node in one network has no reason to serve a requester who pays in the token of another. Exchange across networks today depends on the owner paying for storage separately in each. Fair exchange of payment for data, in which neither party can cheat, is achievable with a smart contract that releases payment against a verifiable proof of delivery, but it adds at least one more ledger transaction to every exchange.

4.7 Design guidelines

The results can be condensed into six guidelines for the design of a blockchain-enabled exchange system.

- Encrypt on the client and place only ciphertext in the storage layer.
- Write to the ledger only a digest, a policy, location identifiers and event records; never the data and never personal identifiers.
- Use a permissioned ledger for a known consortium; on a public chain, batch records under a Merkle root.
- Choose the redundancy scheme from the measured availability of the storage nodes, with a code rate well below that availability, and place copies in more than one network.
- Use a storage-independent digest as the primary key of the registry so that any copy in any network can be verified.
- Treat key management and revocation as the critical security functions, and use proofs of storage where data must be kept for a fixed term.

5. Conclusion

This paper has analysed blockchain-enabled data exchange across decentralized storage systems using protocol constants, published measurements and a probabilistic model. The main findings are as follows.

1. **Only control information belongs on the chain.** Placing one megabyte in Ethereum contract storage would take about 724 million gas, against about 87,300 gas for a registry record. Integrity is not weakened by this separation, since a proof of 704 bytes verifies any chunk of a terabyte file.
2. **Ledger throughput bounds the rate of exchange, not its volume.** A permissioned ledger supports about a hundred times more exchange records than a public chain. The choice depends on whether the parties form a known consortium.
3. **Availability is the weakest property.** In an open network where many peers are unreachable, a moderate-rate erasure code can perform worse than simple replication. A low-rate code such as 29 of 80, or copies in several independent networks, is needed.
4. **Security reduces to key management.** Integrity and non-repudiation are strong. Confidentiality depends on keys, and revocation cannot recall data already decrypted.
5. **Cross-system exchange needs a storage-independent registry.** Different addressing schemes, ledgers and incentives are the remaining obstacles.

Limitations and future work. The evaluation is analytical. The architecture was not implemented, and the figures for gas are computed from the protocol schedule and not measured on a network. The gas limit and fee rules of public chains change over time. The throughput of the permissioned ledger is a reported peak from one study. The availability model assumes independent node failures. Future work should implement the registry and access-control contracts, measure exchange latency across at least two storage networks, and test the availability model against observed retrieval success.

References

1. Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., et al. (2018). Hyperledger Fabric: A distributed operating system for permissioned blockchains. In *Proceedings of the Thirteenth EuroSys Conference* (Article 30). ACM. https://arxiv.org/abs/1801.10228
2. Ateniese, G., Fu, K., Green, M., & Hohenberger, S. (2006). Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Transactions on Information and System Security*, 9(1), 1–30.
3. Belchior, R., Vasconcelos, A., Guerreiro, S., & Correia, M. (2021). A survey on blockchain interoperability: Past, present, and future trends. *ACM Computing Surveys*, 54(8), Article 168.
4. Benet, J. (2014). *IPFS: Content addressed, versioned, P2P file system*. arXiv:1407.3561.
5. Blaze, M., Bleumer, G., & Strauss, M. (1998). Divertible protocols and atomic proxy cryptography. In *Advances in Cryptology: EUROCRYPT 1998* (pp. 127–144). Springer.
6. Daniel, E., & Tschorsch, F. (2022). IPFS and friends: A qualitative comparison of next generation peer-to-peer data networks. *IEEE Communications Surveys & Tutorials*, 24(1), 31–52. https://arxiv.org/abs/2102.12737
7. Herlihy, M. (2018). Atomic cross-chain swaps. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing* (pp. 245–254). ACM.
8. Nakamoto, S. (2008). *Bitcoin: A peer-to-peer electronic cash system*. https://bitcoin.org/bitcoin.pdf
9. Naz, M., Al-zahrani, F. A., Khalid, R., Javaid, N., Qamar, A. M., Afzal, M. K., & Shafiq, M. (2019). A secure data sharing platform using blockchain and interplanetary file system. *Sustainability*, 11(24), 7054. https://doi.org/10.3390/su11247054
10. Protocol Labs. (2017). *Filecoin: A decentralized storage network*. https://filecoin.io/filecoin.pdf

11. Storj Labs. (2024). *Storj: A decentralized cloud storage network framework* (Version 3.1). https://static.storj.io/storjv3.pdf
12. Trautwein, D., Raman, A., Tyson, G., Castro, I., Scott, W., Schubotz, M., Gipp, B., & Psaras, Y. (2022). Design and evaluation of IPFS: A storage layer for the decentralized web. In *Proceedings of the ACM SIGCOMM 2022 Conference* (pp. 688–705). ACM. https://doi.org/10.1145/3544216.3544232
13. 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.
14. Wood, G. (2014). *Ethereum: A secure decentralised generalised transaction ledger* (Ethereum Yellow Paper). https://ethereum.github.io/yellowpaper/paper.pdf