



School of Information Sciences and Technology
Department of Informatics
Athens, Greece

Master Thesis
in
Computer Science

QUADB: A Decentralized Platform for data curation

Nikolaos Lionis

Supervisor: Prof. George Xylomenos
Department of Informatics
Athens University of Economics and Business

Committee: Prof. Vasileios Syris
Department of Informatics
Athens University of Economics and Business

Prof. George C. Polyzos
Department of Informatics
Athens University of Economics and Business

July 25, 2025

Nikolaos Lionis

QUADB: A Decentralized Platform for data curation

July 25, 2025

Supervisor: Prof. George Xylomenos

Athens University of Economics and Business

School of Information Sciences and Technology

Department of Informatics

Mobile Multimedia Laboratory

Athens, Greece

Abstract

Scientific research faces a reproducibility crisis where 70% of researchers cannot reproduce published studies [3], stemming from centralized data infrastructure where platforms silently modify datasets, restrict access, or disappear entirely. This thesis presents **QUADB**, a decentralized platform that systematically integrates five protocols—IPFS, IPNS, FNS, Filecoin, and Lit Protocol—to enable collaborative, persistent, and verifiable dataset management. We establish a protocol composition architecture with design principles for systematic integration, creating emergent capabilities that exceed individual protocol limitations. Our novel threshold-controlled IPNS updates enable m-of-n curator consensus through Lit Protocol, achieving Byzantine fault tolerance with sub-minute latency while maintaining cryptographic security. Through six months of production validation at quadb.xyz, we demonstrate that QUADB achieves practical feasibility despite 5–10× latency compared to centralized systems, providing cryptographic guarantees for permanence, verifiability, and censorship resistance impossible with centralized infrastructure. The platform fundamentally transforms scientific data from fragile corporate assets to resilient community-managed digital commons, enabling truly reproducible science through decentralized coordination mechanisms that resist institutional change and ensure long-term accessibility.

Contents

Abstract	iii
1 Introduction	1
1.1 Research Statement	3
1.2 Data Management Crisis	3
1.3 Core Challenges	4
1.4 Research Questions	4
1.5 Contributions	5
1.6 Thesis Structure	6
2 Background and Related Work	7
2.1 Background	7
2.1.1 IPFS	7
2.1.2 IPNS	8
2.1.3 Filecoin	9
2.1.4 FNS	11
2.1.5 Lit Protocol	11
2.2 Related Work	13
3 System Design and Implementation	17
3.1 Design Goals	17
3.2 System Architecture Overview	18
3.2.1 Storage Layer	19
3.2.2 Naming Layer	21
3.2.3 Crypto Layer	22
3.2.4 Interface Layer	25
3.3 Core Workflows	26
3.3.1 Space Creation	26
3.3.2 Instance Creation	27
3.3.3 Update Process	28
3.4 Quality Signals	29
3.5 Security Considerations	29
3.6 Platform Implementation Details	30
3.6.1 Technology Stack	30

4	Evaluation	33
4.1	Performance Analysis	33
4.2	Economic Analysis	34
4.3	Deduplication Analysis	34
4.4	Validation Results	35
4.5	Security Assessment	36
5	Conclusions and Future Work	39
A	Appendix	41
	Bibliography	47
	List of Acronyms	49
	List of Figures	50
	List of Tables	51
	List of Algorithms	52

Introduction

On October 4, 2021, a single Facebook configuration error did not just cripple social media. It also crippled important research infrastructure used by thousands of scientists around the world. For six hours, epidemiologists could not access COVID-19 dashboards, climate researchers could not access real-time sensor data, and machine learning teams could not retrieve their training datasets [5, 16].

This incident exemplifies a broader systemic vulnerability in our scientific infrastructure. Similar outages at AWS (2017), Cloudflare (2020), and Fastly (2021) show how centralized systems fail when scientists need them most [1, 6, 8]. More recently, the July 2024 CrowdStrike software update affected 8.5 million Microsoft systems worldwide, disrupting airlines, banks, and critical infrastructure for hours [7, 17].

In addition to occasional outages, dependence on centralized platforms creates systematic integrity problems. Large technology platforms silently modify datasets without warning, which can have significant impact on model performance. Changes are often undocumented. Annual storage costs exceeding \$100,000 exclude smaller institutions, while centralized control enables unilateral access restrictions. Platforms can disappear altogether, taking valuable research data with them.

When data can be silently modified, restricted without notice, or permanently lost, empirical research loses its methodological foundation. This undermines both scientific integrity and long-term knowledge preservation, creating a fundamental threat to the reproducibility that forms the cornerstone of scientific methodology.

The movement for a decentralized web offers a different approach. Instead of relying on individual organizations, control is distributed across global computer networks. Different technologies address specific challenges in the areas of storage, naming and collaborative management, but work in isolation. There has been no integrated solution for permissionless data curation with persistence guarantees.

This thesis presents **QUADB**, a platform that combines five decentralized technologies into a unified system for managing digital data and files. Although scientific datasets are our primary use case, QUADB's architecture supports decentralized workflows for any digital content: research datasets, code repositories, documentation, and multimedia files. The platform creates a new paradigm where digital content becomes a permanent, collaborative

digital commons that resists institutional change. QUADB is deployed at quadb.xyz and has been recognized through hackathon awards and funded through Filecoin's retro funding program.

QUADB demonstrates that the integration of complex protocols can provide practical solutions to digital data challenges. The platform enables users to:

- Publish any type of data or files with cryptographic integrity guarantees
- Implement verifiable and permissionless version control
- Maintain complete version histories for evolving content
- Ensure permanent availability through decentralized storage
- Create subscription-based access models for premium content
- Establish private, curated, or open data sharing arrangements

These capabilities work independently of any single institution, fundamentally transforming digital data management from fragmented enterprise dependency to resilient community commons.

1.1 Research Statement

In this thesis we show that systematic integration of decentralized protocols can solve fundamental problems in scientific data management without compromising practical applicability. We find that threshold-driven collaborative management can enable sub-minute updates of datasets with mathematical safety guarantees. This transforms scientific data from fragile corporate assets to durable commons managed by the community. The resulting system offers demonstrable benefits—persistent availability, cryptographic integrity, and resistance to censorship—that justify the measurable cost of performance, enabling the key requirements for reproducible science.

1.2 Data Management Crisis

Scientific data is growing at an incredible pace. Genomics research is projected to require over 40 exabytes of storage by 2025 [13]. Climate data repositories are expected to reach substantial petabyte-scale volumes. Machine learning datasets continue to grow exponentially in size and complexity. Yet centralized systems managing important data have failed multiple times.

Current data management assumptions often prove wrong: institutions may not exist forever, administrators may not be transparent about changes, and free hosting may not remain sustainable. In practice, servers crash, datasets disappear from catalogs, and repositories close when funding ends. This also happens with large datasets such as ImageNet, COCO, and WikiText, where changes are made without notifying researchers.

The key insight from QUADB is that **combining technologies creates new capabilities** that solve concrete challenges:

- **Climate research:** Sensor data is automatically replicated across continents, ensuring global accessibility and redundancy
- **Medical research:** Teams collaboratively maintain patient records with cryptographic audit trails, preserving privacy and enabling verification
- **Machine learning:** Engineers refer to specific versions of data sets that remain permanently accessible, removing barriers to reproducibility
- **Institutional preservation:** Data survives organizational changes, protecting scientific heritage beyond the lifespan of institutions

1.3 Core Challenges

The creation of QUADB required the resolution of four challenges that had hindered previous attempts to create a decentralized data infrastructure. **First, the challenge of mutability:** blockchain and cryptographic systems are designed under the assumption of immutability, but datasets must be updated. The challenge is to enable updates while ensuring security and maintaining a complete history of changes. Traditional approaches force researchers to choose between security and flexibility.

Second, is the challenge of collaborative control: scientific datasets must be approved by multiple experts. It is difficult to create systems in which multiple people can control updates without single points of failure. Current solutions rely on centralized committees or slow blockchain-based consensus, which makes them impractical.

Third, naming challenge: There is a fundamental conflict between security and usability. Cryptographic hashes such as QmR7G...rJa3LX provide security but are useless to humans. It is difficult to create easy-to-remember names while ensuring decentralization. DNS offers user-friendliness but requires central control.

Finally, the economic sustainability challenge: storing scientific data incurs costs over time. If data sets outlive their creators, funding mechanisms must ensure permanent storage without long-term commitments. Current models depend on the longevity of institutions or the charity of companies, both of which are unreliable.

1.4 Research Questions

This thesis addresses three fundamental research questions that guide the development and evaluation of QUADB.

RQ1: Protocol composition feasibility - Can IPFS, IPNS, FNS, Filecoin, and Lit Protocol be integrated to achieve threshold-controlled dataset management while maintaining individual protocol security guarantees?

RQ2: Collaborative update performance - Can threshold cryptography enable m-of-n curator consensus for dataset updates with <60-second latency while resisting Byzantine failures from up to $\frac{n-1}{3}$ malicious participants?

RQ3: Decentralization trade-off quantification - What measurable performance costs (latency, throughput, storage) result from decentralized management, and do the security guarantees (permanence, verifiability, censorship resistance) justify these costs?

1.5 Contributions

In this thesis we make four main contributions that together solve these key challenges.

First, we present a protocol composition architecture that allows heterogeneous systems to work together harmoniously. Through carefully designed interfaces using IPFS CIDs and FNS domains, the protocols achieve loose coupling while integrating seamlessly. These patterns can be reused for other protocols, such as Arweave and Ceramic. The robustness of the architecture has been validated by a production implementation on quadb.xyz.

Second, we present threshold-driven IPNS, a new algorithm that uses the Lit protocol to manage IPNS private keys among distributed nodes. This allows for m-of-n threshold signatures for record updates—for example, 3 out of 5 curators must approve changes. The system achieves an update latency of less than one minute while providing Byzantine fault tolerance without compromising IPNS private keys to the curators.

Third, we provide a complete system implementation with over 15,000 lines of TypeScript production code, fully open-sourced for community review. The platform includes a web interface for data management and smart contracts on FEVM for metadata anchoring, curator configuration management, quality reporting, FNS domain management, and subscription data access.

Fourth, we establish a theoretical framework that quantifies the performance-decentralization tradeoffs associated with blockchain systems. Our analysis shows that the performance penalty is a fundamental property of decentralization and not an implementation artifact. Performance depends on the underlying network (currently Filecoin), but because the protocol is chain-agnostic, it can be deployed on any EVM-compatible network.

1.6 Thesis Structure

Chapter 2: Background and Related Work

Examines the development of decentralized technologies from the first peer-to-peer systems to modern blockchain protocols. Analyzes the reasons for the failure of previous attempts at decentralized data management and identifies the missing elements that QUADB can provide.

Chapter 3: The QUADB Architecture

Describes in detail the technical design and implementation of QUADB. Shows how five independent protocols achieve seamless integration, introduces the new threshold-controlled IPNS algorithm, and demonstrates the system works using real-world examples.

Chapter 4: Evaluation and analysis

Quantifies QUADB's performance across multiple dimensions: latency, throughput, storage costs, and decentralization metrics. Presents results from user studies and analyzes real-world usage patterns from the live deployment.

Chapter 5: Conclusions and Future Vision

Synthesizes key findings and the path forward. Discusses how QUADB's architectural patterns can transform not just scientific data management, but the broader landscape of decentralized applications.

Background and Related Work

This chapter examines the individual decentralized networks and protocols used by QUADB: IPFS, IPNS, Filecoin, FNS, and Lit Protocol and explain their capabilities and limitations. We then review existing data management platforms to identify the gaps that QUADB fills and show how the combination of these technologies creates a decentralized solution for data curation.

2.1 Background

QUADB combines proven decentralized technologies, leverages their strengths, and overcomes their individual limitations to create a unified platform. Understanding these building blocks is crucial to understanding the architectural innovation of QUADB.

2.1.1 IPFS

The InterPlanetary File System (IPFS) provides decentralized file storage and sharing without central servers. Instead of storing files in central locations controlled by individual organizations, IPFS distributes files across a global network of participating computers.

IPFS works by assigning each file a unique digital fingerprint, known as a Content Identifier (CID). This digital fingerprint is based on the content of the file and identifies the file by helping to locate it on the network. Unlike traditional web addresses, which point to a specific server, IPFS addresses point to the content itself, regardless of where it is stored.

Despite its innovations in content addressing, IPFS has systematic limitations that prevent its standalone use for managing scientific data

Persistence fragility: The availability of content depends entirely on the voluntary participation of nodes without economic incentives. Research by Trautwein et al. [23] shows that 50% of content is no longer available within 24 hours of upload due to nodes being abandoned. This leads to fundamental reliability issues for scientific datasets that require long-term accessibility.

Performance limitations: IPFS has 5–10× higher latency compared to CDNs, with content discovery times varying between 3 and 45 seconds depending on network topology. Distributed hash table (DHT) routing suffers from scalability limitations at more than 10⁶ nodes, leading to bottlenecks for large-scale scientific collaboration.

Economic sustainability gaps: The lack of built-in incentives for providing storage space leads to a tragedy of the commons, where storage costs are externalized while the benefits are shared, resulting in systematic underinvestment in network infrastructure and content persistence.

2.1.2 IPNS

While IPFS is great for storing immutable content, scientific data sets need to evolve over time. This leads to the “mutability problem”: how can researchers update data sets while keeping the security benefits of content addressing?

The InterPlanetary Naming System (IPNS) [21] fixes this fundamental limitation. IPFS provides immutable, content-based objects with permanent links, universal integrity checking, and caching capabilities. However, it is not capable of managing content that needs to be updated over time. IPNS solves this problem by providing mutable references to dynamic content using the concept of the Self-Certifying File System (SFS) concept [15, 14]. The system creates a mutable namespace for each user based on their public key hash, which is accessible via the path `/ipns/ipns-id`. Within this namespace, users can publish cryptographically signed objects that refer to different versions of their content and disseminate them to the network.

The fundamental innovation of IPNS is its ability to maintain a stable and consistent point of reference while allowing the underlying content to evolve. This design preserves all the advantages of IPFS’s immutable, content-based system while introducing the necessary level of mutability required for dynamic applications.

However, IPNS introduces critical limitations that restrict its applicability for collaborative management of scientific data:

Update propagation delays: IPNS operations over the DHT can be slow due to the need to find multiple records across nodes to ensure the latest version is retrieved. The IPNS specification [12] notes that publishing and resolving IPNS names using the DHT involves round trips to multiple nodes, creating latency in update propagation and temporary inconsistency windows where different nodes may serve different dataset versions.

Single key control bottlenecks: Each IPNS data record requires exactly one private key holder, which prevents collaborative management. This centralized control model contradicts the requirements of scientific collaboration, where multiple researchers need shared authority over data sets. Compromising or losing the key permanently disables updates to the data sets, with no recovery mechanisms in place.

Tensions between security and usability: IPNS names (e.g., k2k4r8n7v . . .) are cryptographically secure but unusable by humans. Integration with DNS provides usability but reintroduces centralized dependencies that undermine the benefits of decentralization.

2.1.3 Filecoin

Both IPFS and IPNS rely on the voluntary participation of network nodes to store and provide data. This leads to a critical vulnerability: there is no guarantee that important scientific data sets will remain available in the long term. If nodes disconnect or terminate their participation, data may be lost. Filecoin addresses this persistence problem by extending IPFS with economic incentives for reliable, long-term data storage, thereby creating a decentralized marketplace [4]. The network uses a blockchain infrastructure with a native token(FIL) to create an open marketplace where users can rent storage space from storage providers.

The security and reliability of Filecoin are based on two fundamental cryptographic mechanisms. **Proof-of-Replication (PoRep)** ensures that storage providers have actually stored replicas of the data and are not just declaring their storage capacity [9]. In addition, **Proof-of-Spacetime (PoSt)** provides continuous verification that the data is maintained over extended periods of time [10]. Together, these mechanisms create a verifiable market with performance guarantees while promoting honest behavior [20].

These cryptographic proofs also form the foundation of Filecoin's consensus mechanism, enabling network participants to agree on the shared ledger state without centralized intermediaries. The consensus protocol operates through a Proof of Stake-like system that calculates each miner's influence based on their contributed storage capacity, termed storage power.

These cryptographic proofs also form the basis for Filecoin's consensus mechanism, which enables network participants to agree on the status of the shared registry without a central intermediary. The consensus protocol works via a system similar to Proof of Stake, which calculates the influence of each miner based on the storage capacity provided, known as storage power.

The network employs Expected Consensus to select block producers through a deterministic but unpredictable process that maintains security while ensuring fair participation. At each epoch, the protocol secretly elects a group of leaders from among the storage providers who meet the minimum storage capacity requirements, while giving more weight to those with higher storage power. These elected leaders generate the next set of blocks and receive FIL token rewards, with each epoch potentially containing zero or more blocks organized into tipsets that form the blockchain structure.

Despite solving storage incentivization, Filecoin exhibits architectural limitations that prevent standalone deployment for scientific data management:

Integration isolation: Filecoin operates independently from IPFS content addressing and naming systems, requiring manual coordination between storage deals and content identification. No built-in mechanisms exist for linking Filecoin storage contracts to IPFS CIDs or IPNS records, creating operational complexity for integrated workflows.

Economic barriers to entry: Storage deals require significant FIL token stakes (typically 1-10 FIL minimum) and complex negotiation processes unsuitable for small-scale research projects. Gas fees for storage operations can exceed \$10-100 per transaction during network congestion, creating cost barriers for budget-constrained research institutions.

Management mechanism absence: Filecoin focuses exclusively on storage provision without addressing dataset management, versioning, access control, or collaborative curation. Storage providers have no incentives for maintaining dataset quality, version history, or coordination compliance beyond basic storage availability.

Retrieval performance limitations: While Filecoin guarantees storage, it provides limited guarantees for retrieval performance. Storage providers may impose bandwidth limitations, latency constraints, or access restrictions that degrade dataset usability for time-sensitive research applications.

2.1.4 FNS

Filecoin Name Service (FNS) is a decentralized naming system that maps human-readable identifiers to machine-resolvable records such as blockchain addresses, content hashes, and metadata URIs [25]. Operating on the Ethereum blockchain, FNS eliminates reliance on centralized DNS authorities, offering censorship resistance, cryptographic verifiability, and programmability through smart contracts.

FNS namespaces are hierarchical and rooted in ownership of top-level domains (TLDs), allowing for structured delegation. Each name is managed by a smart contract that specifies resolver logic and access control policies. These resolvers can associate names with IPFS CIDs, IPNS records, or other identifiers, facilitating integration with decentralized content networks.

2.1.5 Lit Protocol

Lit Protocol provides decentralized infrastructure for secure, programmable cryptographic key management [22]. The protocol uses a distributed network of independent nodes that employ threshold cryptography [26] and hardware isolation.

The security model relies on Trusted Execution Environments (TEE), specialized hardware that isolates sensitive cryptographic keys from system operations. This hardware-based isolation prevents unauthorized access while maintaining network distribution. Secrets are split using threshold secret sharing and distributed across multiple nodes. Cryptographic operations only occur when a predefined threshold of nodes reaches consensus.

The protocol includes Access Control Lists (ACL) for granular control over cryptographic operations. ACLs enable complex conditions for content encryption and decryption based on blockchain states, including token balances, network conditions, and smart contract interactions. This programmable access control ensures only users meeting specific conditions can decrypt content.

Lit Protocol also offers Lit Actions—custom JavaScript functions that run securely within the decentralized network. These cryptographically verified functions enforce complex rules and integrate with ACLs. Lit Actions can serve as oracles, bringing off-chain access rules into blockchain networks and enabling integration with external protocols.

Storacha

Storacha (formerly web3.storage) represents a paradigm shift in decentralized storage by providing a cloud-like interface that seamlessly integrates IPFS, Filecoin, and IPNS protocols. The platform maintains decentralized principles while offering service guarantees that ensure secure, verifiable, and persistent data storage through familiar user experiences.

The platform's core strength lies in its comprehensive service guarantees that span multiple decentralized protocols. Through IPFS integration, Storacha provides robust content addressing by pinning files to IPFS networks and distributing them across public nodes, ensuring global accessibility without relying on centralized infrastructure. The platform's Filecoin integration creates verifiable storage deals that users can permissionlessly audit, providing cryptographic proof that their files are included in long-term storage commitments.

Storacha's IPNS support enables mutable references for evolving datasets, allowing users to maintain consistent access points for dynamic content while preserving the benefits of content-addressed storage. The platform implements User Controlled Authorization Networks (UCAN) [24] to provide sophisticated access control mechanisms, including client-side authorization, fine-grained permissions, secure delegation capabilities, and automated service health monitoring.

Eliminating the complexity of interacting with multiple decentralized protocols, Storacha provides a unified interface that combines traditional expectations of cloud computing with the fundamental benefits of a decentralized infrastructure. Users gain access to censorship resistance, data sovereignty, and cryptographic verifiability without requiring deep technical knowledge of the underlying decentralized systems.

2.2 Related Work

Having examined the individual technologies that QUADB integrates, we now turn to existing platforms and academic research in decentralized data management. Several platforms have attempted to address the challenges of decentralized data curation and marketplaces, but significant gaps remain in practical, integrated solutions. This section reviews these efforts and identifies how QUADB's approach differs.

Data Management Platforms

Ocean Protocol [19] focuses on data marketplaces with privacy-preserving analytics, enabling data monetization through blockchain-based access control. While Ocean addresses data commerce, it lacks versioning capabilities and depends on centralized storage infrastructure. The platform emphasizes data exchange rather than collaborative management, missing the multi-curator model essential for scientific datasets.

Ceramic Network [18] provides mutable data streams with decentralized identity, offering programmable data sovereignty. However, Ceramic's focus on application data rather than large-scale datasets creates scalability limitations. The platform lacks integration with storage incentive mechanisms, relying on voluntary pinning services that cannot guarantee long-term persistence.

Arweave [2] offers permanent storage through novel economic incentives, addressing data persistence challenges. However, Arweave's "write-once" model prevents collaborative dataset evolution, and its lack of mutable references requires external naming systems. The platform provides storage without coordination mechanisms for multi-party dataset curation.

QUADB Contributions

QUADB systematically addresses the identified limitations of existing technologies through architectural innovations. These create previously impossible capabilities:

(1) Systematic protocol composition methodology: While individual protocols suffer from isolation (IPFS lacks persistence incentives, IPNS lacks collaborative record updates, Filecoin lacks naming integration, FNS lacks storage capabilities, Lit Protocol lacks application integration), QUADB establishes the first comprehensive framework for composing these heterogeneous systems. The loose coupling through standardized interfaces preserves individual protocol security properties while creating emergent system capabilities impossible with any single technology.

(2) Threshold-controlled collaborative management: Existing platforms fail to solve the fundamental tension between security and collaboration—IPNS requires single-key control, FNS supports only ownership-based access, and traditional blockchain voting is too slow for practical workflows. QUADB introduces the first practical implementation of threshold-controlled IPNS records through Lit Protocol integration, enabling m-of-n curator consensus with sub-minute latency while maintaining Byzantine fault tolerance acting as a multisignature permissionless secure serverless function. This solves the collaborative dataset evolution problem that no existing platform addresses.

(3) Economic sustainability without barriers: While Filecoin creates cost barriers and Ocean Protocol focuses on data monetization, QUADB demonstrates sustainable scientific data governance through minimal transaction costs (0.001-0.003 FIL per operation) and storage persistence guarantees without requiring complex storage deal negotiations. The platform abstracts economic complexity while providing cryptographic persistence guarantees.

(4) Production-scale empirical validation: Unlike academic research that remains theoretical or platforms that provide limited real-world validation, QUADB demonstrates practical feasibility through production deployment with comprehensive real-world usage data. This represents the first comprehensive empirical validation of multi-protocol integration for scientific data management, providing performance data and usability insights unavailable from existing platforms.

(5) Comprehensive dataset lifecycle management: Existing platforms address single aspects—storage (IPFS, Arweave), naming (FNS), or data marketplace (Ocean Protocol)—but no integrated solution exists for complete dataset management. QUADB provides the first platform combining content integrity, version control, collaborative coordination,

human-readable naming, and economic sustainability in a unified architecture suitable for scientific workflows and extendable to other use cases.

Most critically, QUADB demonstrates that the fundamental limitations of individual decentralized protocols can be systematically addressed through careful architectural composition rather than requiring new protocol development. This approach enables practical deployment using mature, battle-tested protocols while achieving novel capabilities through integration innovation.

This chapter outlined the foundational tools QUADB builds upon. The next chapter presents the architecture, implementation, and workflows of QUADB in detail.

System Design and Implementation

This chapter presents the design and implementation of QUADB, explaining how the technologies discussed in Chapter 2 are combined into a unified system. The structure of this chapter follows a layered approach, providing a clear overview of the system's goals, architecture, and core workflows.

3.1 Design Goals

QUADB was designed as an intuitive, open, secure and decentralised platform for managing scientific data. The system aims to combine ease of use with a high level of data protection so that researchers can exchange, update and store data sets securely over a long period of time. The main design objectives are as follows:

- **Decentralization:** No single group or individual controls access or changes to records, removing central points of failure.
- **Controlled Evolution:** Datasets maintain a permanent record of all versions, while allowing new versions to be published through secure storage, updateable links, and group approval for changes.
- **Reliable Storage:** Datasets remain available over time because storage providers are rewarded for keeping them safe, using systems such as Filecoin and automatic storage checks.
- **Community Quality Signals:** Users can show trust in dataset versions by staking tokens, creating quality signals based on real participation.

These principles make QUADB a decentralized, sustainable, and accessible platform for collaborative scientific data management.

3.2 System Architecture Overview

QUADB's architecture is based on the integration of existing blockchain protocols, each maintaining its own security properties while contributing to the system's goals. This approach allows components to evolve independently. Figure 3.1 provides a high-level overview of the system.

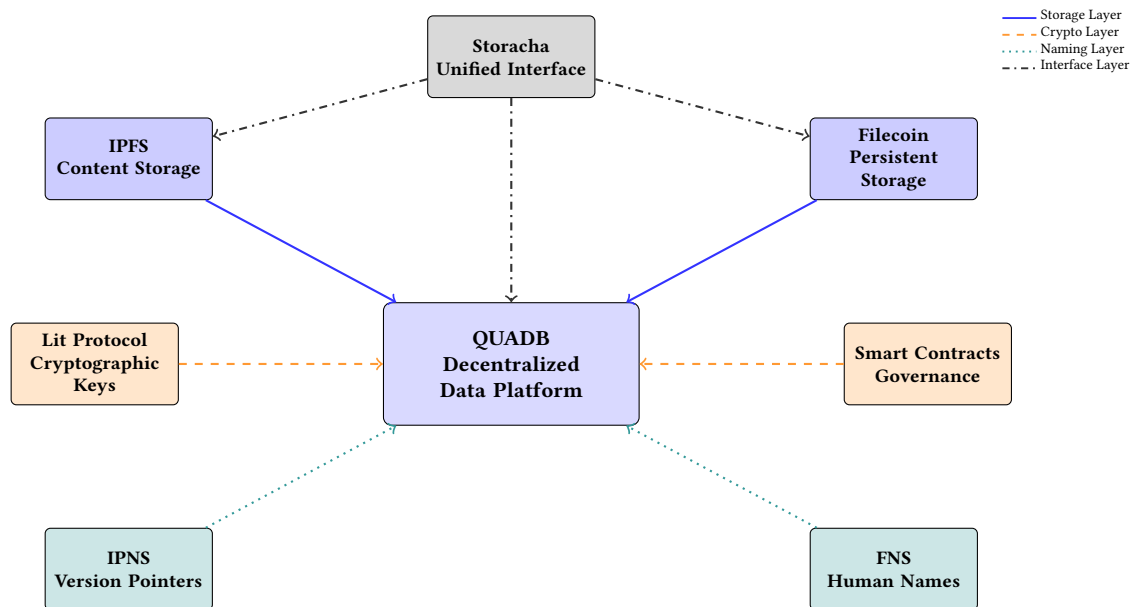


Fig. 3.1.: QUADB system architecture across the four interconnected layers

The architecture of QUADB organizes five decentralized protocols into four layers:

- **Storage Layer:** Manages data storage using IPFS for content addressing and Filecoin for long-term retention.
- **Naming Layer:** Provides updatable and human-readable references using IPNS and FNS.
- **Crypto Layer:** Manages access control and permissionless updates using Lit Protocol and smart contracts.
- **Interface Layer:** Simplifies protocol coordination through Storacha, offering a unified interface.

3.2.1 Storage Layer

The Storage Layer forms the foundation of QUADB's data management, ensuring that datasets are stored in a decentralized, verifiable, and persistent manner. This section describes how QUADB leverages IPFS, Filecoin, and Storacha to provide robust storage guarantees and efficient data organization. The Storage Layer integrates IPFS content-addressing with Filecoin's persistence guarantees, coordinated by Storacha. IPFS provides verifiable and immediate access to content, while Filecoin ensures long-term retention through economic incentives and proof of storage. Storacha coordinates these systems, allowing users to interact with both protocols seamlessly. Fig 3.2 shows the storage layout. In QUADB, each dataset is organized under a root CID (ρ) with a standardized hierarchical structure that stores all essential components in separate, easily indexable subdirectories:

- **/data:** The dataset files themselves.
- **/gov:** Governance information required for verifiable updates, such as the Lit Action configuration and curator rules .
- **/persist:** Filecoin deal information, enabling verification of storage permanence by linking to the relevant Filecoin storage contracts for the stored files.
- **/meta:** Arbitrary metadata about the dataset, supporting discovery and descriptive information.

This structure enables efficient access, supports verifiable updates, and allows for independent retrieval and validation of each component. The separation of these elements ensures that data, governance, storage proofs, and metadata can be managed and indexed independently, supporting robust and transparent dataset management across layers storage layout is also formalized in Algorithm 1.

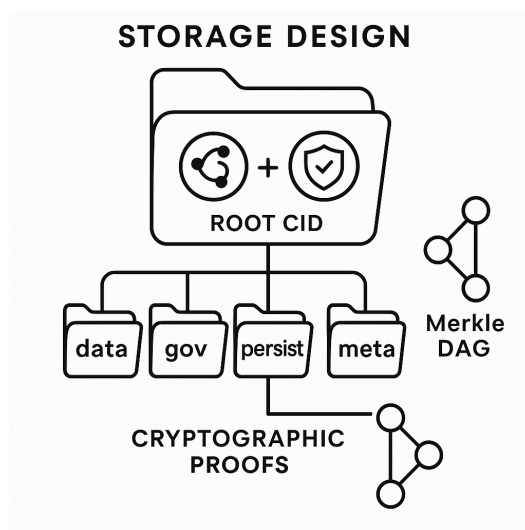


Fig. 3.2.: Storage design in quadb

Data Deduplication

QUADB implements data deduplication across dataset versions, leveraging IPFS's content-addressed storage to improve storage efficiency. The system uses file-type-aware deduplication algorithms:

- **Merkle DAG-based deduplication:** IPFS's Merkle DAG structure enables deduplication at the block level. Identical blocks in different versions share the same content identifier (CID), resulting in automatic deduplication without extra storage costs.
- **File-type optimization:** Text files (CSV, JSON, XML) benefit from row-level deduplication, while binary files (images, archives) use block-level strategies. This adaptive approach maximizes efficiency while maintaining performance [11].
- **Future optimization:** The architecture supports advanced deduplication through pre-calculated file DAGs, enabling real-time analysis and predictive storage optimization.

This deduplication mechanism reduces storage costs for evolving datasets while preserving complete version history, making long-term retention economically viable.

3.2.2 Naming Layer

The Naming Layer addresses the challenge of referencing datasets in a mutable yet reliable way. Here, we explain how QUADB combines IPNS and FNS to provide both stable, updatable identifiers and human-readable names, supporting both technical and user-friendly access to data.

Each dataset instance receives a unique IPNS name, generated through the w3up service, providing a stable identifier that can be updated to point to new content versions. FNS domains give datasets human-readable names, improving usability. When users reference a dataset through its FNS domain, the system resolves this to an IPNS name, which in turn points to the current IPFS CID. This multi-layer naming system provides both stability and flexibility, allowing datasets to evolve while maintaining permanent access to all historical versions.

QUADB leverages the Filecoin Name Service (FNS) to create a hierarchical namespace of data repositories. The root name is owned and managed by the QUADB smart contract, and each node in the namespace represents a repository of data repositories. This structure supports extensibility, as anyone can add new subdomains to extend the namespace. The use of FNS subdomains enables organized, human-readable, and decentralized management of datasets.

Figure 3.3 illustrates the tree-like structure of the namespace, making navigation intuitive and scalable.

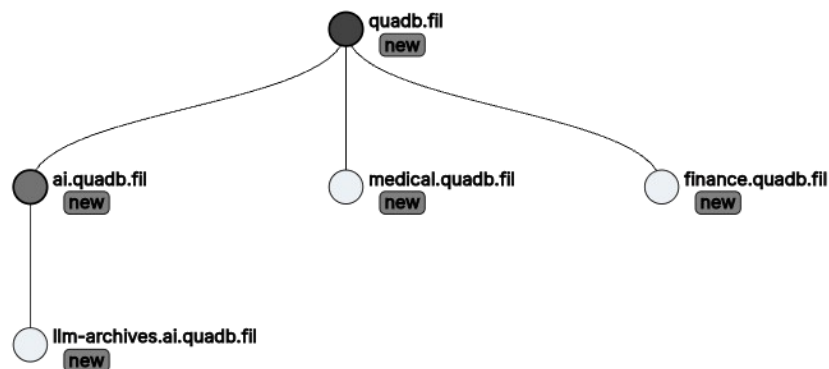


Fig. 3.3.: Tree-like namespace structure in QUADB using FNS subdomains. Each node represents a data repository, and the hierarchy can be extended by anyone, supporting organized and scalable data management.

The QUADB explorer interface (see Figure A.1) further enhances usability by providing an intuitive way to browse and interact with the namespace.

3.2.3 Crypto Layer

The Crypto Layer is responsible for secure access control and collaborative governance of dataset updates. This section details how QUADB uses threshold cryptography and smart contracts to enable permissionless, multi-party control over dataset evolution. Lit Protocol enables m-of-n consensus for dataset updates, requiring multiple curators to approve changes. The smart contract, deployed on the Filecoin EVM, serves as the authoritative registry for data instances and their governance parameters that are used from the Lit Actions to validate and perform the updates.

Permissionless Updates with Lit Actions

QUADB's coordination system enables multiple researchers to collaboratively control dataset evolution without single points of failure. Lit Protocol's threshold cryptography implements m-of-n consensus for updates. The integration addresses three critical aspects:

- **Secure key management:** Dataset-specific IPNS keys are generated and encrypted by the Lit Protocol network, ensuring no single entity has unencrypted access.
- **Permissionless updates:** Lit Actions are executable scripts stored on IPFS, identified by immutable CIDs. Only the Lit Protocol network can execute these scripts, enforcing consensus requirements.
- **Conditional execution:** The Lit Action verifies that a threshold number of curators have signed their approval. This occurs within the Lit Protocol's secure environment, ensuring consensus rules cannot be bypassed.

To protect against replay attacks, each update message includes both the new IPNS pointer and the current IPNS sequence number. The Lit Action checks that the sequence number in the update is strictly greater than the previous value. This prevents attackers from submitting previously signed messages to revert the dataset to an older version, ensuring that only forward progress is possible and that permissionless updates remain secure. Figure 3.4 illustrates how Lit Actions enable permissionless, threshold-based updates to IPNS records without exposing private keys.

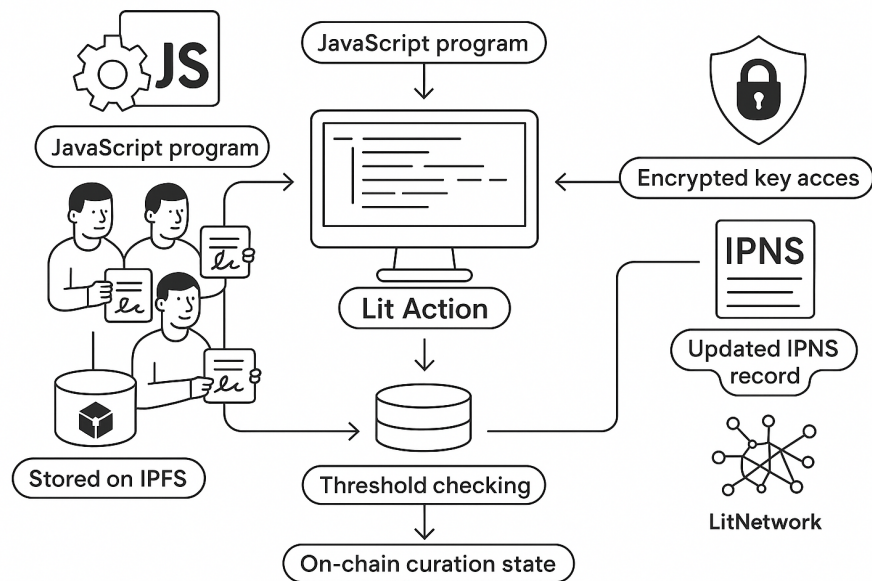


Fig. 3.4.: Permissionless updates: Lit Actions enable threshold-based updates to IPNS records. Curators sign approvals, which are verified by the Lit Protocol network. The process updates the IPNS record without exposing private keys.

The smart contract records the instance’s IPNS name, the IPFS CID of the Lit Configuration JSON, and the list of authorized curators and signature thresholds. This establishes the trust foundation for all future modifications.

Lit Action Implementation

The implementation of QUADB Lit Actions required a customized integration to enable the desired functionality. Lit Actions operate within a sandboxed JavaScript environment that supports only a subset of JavaScript capabilities. Custom library compilation was necessary to support three essential functions within the Lit Protocol environment: signature verification and address recovery for curator authentication, smart contract interaction for governance rule validation, and IPNS network communication for record updates. These components were compiled to be compatible with the Lit Protocol’s restricted execution environment while maintaining the cryptographic security properties required for permissionless dataset updates.

Lit Action Factory: The Lit Action Factory purpose is to create a lit action for each dataset by taking the base compiled code and replacing the datasetID to enforce the access control rules for the dataset. While each datasetID can be deterministically generated this is not introducing further security concerns.

Access Control Models

QUADB's smart contract architecture supports multiple access control models to accommodate diverse data sharing requirements:

Open Instances provide unrestricted access to any user, ideal for public datasets, open-source code, or community resources that benefit from maximum accessibility.

Private Instances implement NFT-based gated access through the GatedInstance contract. Only users holding specific NFTs can access the content, enabling controlled collaboration among defined research teams or organizations. There is also the admin who is able to grant/revoke access to specific users. That brings considerations as one entity can manage the access control but that could be solved by using a multisig contract.

Subscription-Based Instances utilize the SubscriptionNFTs contract to provide time-based access through economic participation. Users purchase monthly subscriptions to access premium datasets or commercial research content, with 99% of subscription fees forwarded to content creators.

Hybrid Models combine multiple access controls—for example, private instances with subscription options allow both team members (via NFT membership) and external subscribers (via paid access) to utilize the same content under different terms.

Each instance access control model is anchored on-chain, encryption and decryption rules are enforced by the Lit Protocol which relies on custom rules based on the access control model. This flexible access control system enables QUADB to support diverse use cases from open scientific collaboration to commercial data monetization while maintaining the same underlying decentralized infrastructure.

3.2.4 Interface Layer

The Interface Layer abstracts the complexity of interacting with multiple decentralized protocols. The platform architecture prioritizes modularity to facilitate integration with evolving decentralized protocols and storage providers. The interface layer currently utilizes Storacha for IPFS pinning, Filecoin storage deal management, and IPNS record operations, but the abstraction layer enables seamless integration of alternative storage providers and protocols.

This modular approach ensures that QUADB can adapt to technological developments in the decentralized storage ecosystem while maintaining consistent functionality for end users. The abstraction layer provides standardized interfaces for storage operations, naming services, and cryptographic functions, enabling protocol substitution without requiring application-level modifications.

3.3 Core Workflows

To illustrate the practical operation of QUADB, this section walks through the core workflows involved in creating, updating, and managing datasets. These workflows demonstrate how the system's layered architecture supports real-world use cases.

This section details the practical implementation of QUADB's core workflows.

3.3.1 Space Creation

The creation of a new dataset instance in QUADB begins with establishing an FNS Space. This process creates the organizational subspace that contributes to quadb's hierarchical namespace. Figure 3.5 illustrates this workflow.

The hierarchical naming architecture implements a tree-based namespace organization where `quadb.fil` serves as the root domain. This design enables systematic dataset categorization through subdomain structures (e.g., `11m.quadb.fil` for machine learning datasets, `genomics.quadb.fil` for biological data) each of which can be conceptualised as a repository of data repositories in QUADB namespace.

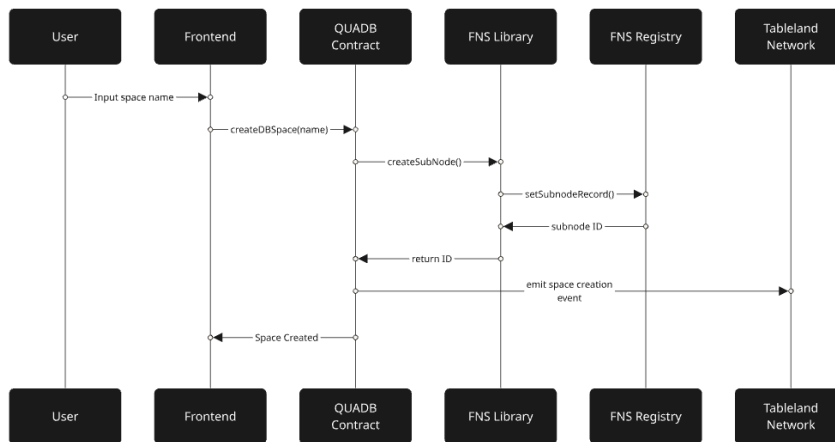


Fig. 3.5.: QUADB Space Creation Sequence Diagram

3.3.2 Instance Creation

Creating a new dataset instance in QUADB is a multi-step process that is streamlined for the user. The sequence diagram in Figure 3.6 shows the main steps involved in this process.

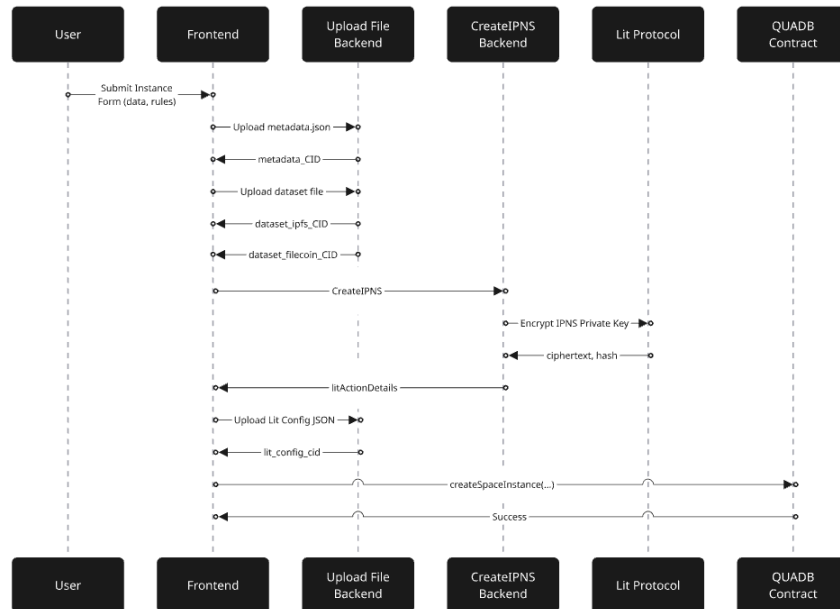


Fig. 3.6.: Dataset Instance Creation Sequence Diagram

The key steps for creating a dataset instance are:

1. Upload the dataset files to IPFS and Filecoin for decentralized storage.
2. Create an IPNS record for the dataset instance.
3. Generate a LitAction for the dataset, which is used to update the IPNS record.
4. Encrypt the IPNS private key using the Lit Protocol, so only the LitAction can decrypt it after validating the required inputs.
5. Combine the results of the previous steps into a single storage layout and upload it to IPFS, as described in Section 3.2.1.
6. Anchor the dataset metadata and curation rules on-chain.
7. Register the dataset instance on the QUADB smart contract.

This process ensures that each dataset instance is securely stored, versioned, and governed in a decentralized manner. By combining decentralized storage, threshold-based key management, and on-chain governance, QUADB achieves both data integrity and controlled mutability. The use of smart contracts and cryptographic protocols allows multiple curators to collaboratively manage updates, while preserving a complete and tamper-evident version history. For technical details on the IPNS record creation protocol, see Algorithm 3.

3.3.3 Update Process

Updating a dataset instance in QUADB is a collaborative process that ensures only authorized changes are made. The sequence diagram in Figure 3.7 illustrates the main steps involved.

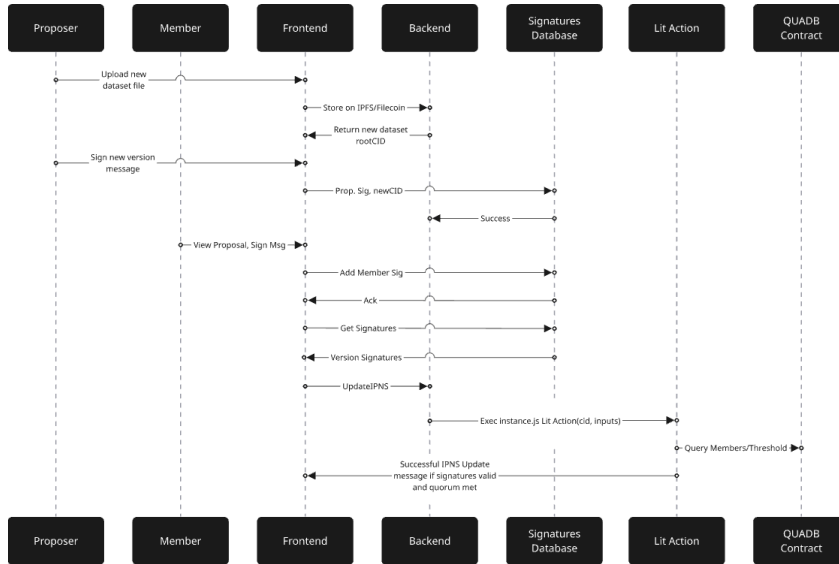


Fig. 3.7.: Dataset Instance Update Sequence Diagram

The key steps for updating a dataset instance are:

1. The user proposes a new version of the dataset.
2. The user collects signatures from the curators.
3. If the threshold is reached, the user calls the LitAction to update the IPNS record.
4. The user passes the new CID for IPNS to point to the new version of the dataset, along with the signatures of the curators.
5. The LitAction queries the smart contract to get the curators and the threshold required to update the IPNS record. It also validates the sequence number of the IPNS record, creates the message that should have been signed, and validates the signatures to recover the addresses that signed.
6. On successful validation, the LitAction decrypts the IPNS private key and updates the IPNS record with the new pointer.

This process ensures that updates are only made when the required number of curators approve the change. It maintains the integrity and security of the dataset, while allowing for collaborative and permissionless updates. For technical details, see Algorithm 2.

3.4 Quality Signals

QUADB incorporates an experimental quality signaling mechanism that explores stake-weighted voting for dataset assessment. Users can economically express confidence in datasets through staking, creating market-driven quality indicators that complement traditional peer review. The experimental framework investigates whether economic incentives can effectively motivate accurate quality evaluation while deterring gaming behaviors. Implementation combines on-chain stake management with off-chain analytics to generate comprehensive quality metrics spanning technical accuracy, documentation completeness, and community feedback.

This experimental approach represents an ongoing investigation into cryptoeconomic dataset governance, with findings contributing to the broader understanding of decentralized quality assessment mechanisms.

3.5 Security Considerations

The QUADB security model is based on several properties:

- **Content Integrity:** IPFS CIDs ensure information integrity through cryptographic hashing.
- **Access Control:** Lit Protocol's threshold encryption requires m-of-n authorization for IPNS updates. This ensures that only authorized curators can update the IPNS record, preventing unauthorized changes, while not exposing private keys to anyone and handling replay attacks.
- **Availability:** Distributed storage achieves high availability through redundancy and economic incentives.

3.6 Platform Implementation Details

This section presents the technical implementation architecture of QUADB, detailing the software stack, deployment infrastructure, and custom protocol adaptations that enable the system's decentralized functionality. The implementation demonstrates how established web technologies can be effectively integrated with emerging decentralized protocols to create a production-ready platform. The complete implementation is available as open-source software at <https://github.com/nijoe1/quadb>.

3.6.1 Technology Stack

QUADB is implemented as a full-stack TypeScript application, leveraging modern web development frameworks to provide an accessible interface to complex decentralized infrastructure. The platform architecture consists of three primary components:

The frontend employs React 18 with Next.js 14, providing server-side rendering capabilities and optimized performance for decentralized content access. The user interface utilizes Tailwind CSS for responsive design and the shadcn/ui component library for consistent, accessible UI elements. This combination ensures cross-platform compatibility while maintaining the responsive design patterns expected in modern web applications.

The backend infrastructure leverages Next.js API routes deployed as serverless functions, enabling scalable processing of computationally intensive operations. While the system architecture supports complete client-side operation, the backend provides optimization for resource-intensive tasks such as cryptographic operations and large file processing, significantly improving user experience without compromising decentralization principles.

The blockchain integration layer utilizes Viem and Wagmi libraries for type-safe smart contract interactions with the Filecoin EVM. This approach ensures robust communication between the web application and on-chain governance mechanisms while maintaining compatibility with the broader Ethereum ecosystem.

The interactions with the Lit Network are handled using the Lit Protocol SDK. The Lit Protocol SDK is a library that allows you to interact with the Lit Network. It is used to encrypt and decrypt content with custom rules such as the ACL rules described in Section 3.2.3. As well using lit actions. QUADB uses lit actions to perform the permissionless updates to IPNS records while also using custom ACL rules that are based on the access control model of the dataset anchored on the smart contract.

To create deploy and test QUADB smart contracts we used the hardhat framework. Hardhat is a development framework for Ethereum that provides a comprehensive set of tools for developing, testing, and deploying smart contracts. It is used to create, deploy, benchmark and test QUADB smart contracts.

Protocol Integration Overview

The QUADB architecture systematically integrates five decentralized protocols, each of which plays a distinct role in creating a comprehensive data management system that surpasses the capabilities of individual technologies:

IPFS: Content Addressing and Hot Storage forms the basis for decentralized file storage and retrieval. IPFS acts as a hot storage layer, enabling instant access to content via cryptographic addressing and distributing files across a global network. Its content-addressing nature ensures data integrity through immutable content identifiers (CIDs) that change whenever the content changes.

IPNS and Lit Protocol: Verifiable and secure mutability addresses the fundamental challenge of enabling collaborative evolution of data sets while maintaining cryptographic security. IPNS provides mutable pointers to evolving content, while Lit Protocol's threshold encryption enables m-of-n consensus among curators for updates. This combination creates secure and verifiable mutability where the evolution of datasets requires approval from multiple parties without exposing private keys to individual curators.

Filecoin: Long-term Persistence and Cold Storage ensure the survival of datasets beyond their creators through economic incentives for storage providers. Filecoin acts as a cold storage layer, providing cryptographic proof of storage and guarantees of recovery. Integration with IPFS creates a complete storage lifecycle in which data seamlessly transitions from hot storage (immediate access) to cold storage (long-term archiving).

FNS: Decentralized, human-readable naming bridges the gap between cryptographic security and usability by providing human-readable domain names that resolve to data records. This eliminates the need for users to interact directly with cryptographic hashes, preserving decentralized control.

Storacha: Unified Integration Layer abstracts the complexity of multi-protocol coordination and provides a simplified interface for IPFS pinning, Filecoin storage contracts, and IPNS record management. This integration layer enables QUADB to leverage multiple protocols without requiring users to understand the underlying technical complexity.

This composition creates new properties that would not be possible with individual protocols: content-based records that are jointly managed, permanently stored, human-readable, and economically sustainable, transforming digital data from fragile corporate assets into shared digital goods that are resiliently managed by the community.

Summary

This chapter presented the design rationale, architecture, and implementation details of QUADB. By composing decentralized technologies in a modular way, the system enables reproducible, collaboratively governed, and economically incentivized dataset publication and maintenance. The next chapter evaluates the system's functionality, performance, and feasibility.

Evaluation

This chapter evaluates QUADB through comprehensive analysis of its real-world deployment and performance. The platform received external validation through first prize awards at ETHBrussels and Filecoin hackathons, plus acceptance into the Filecoin Grants Program.

We evaluate QUADB's functionality, performance trade-offs, security properties, economic costs, and comparison with alternatives through production deployment analysis, performance benchmarking, security assessment, and comparative evaluation.

4.1 Performance Analysis

QUADB's performance analysis reveals the fundamental trade-off between decentralization benefits and speed. QUADB operations require more time than centralized systems but provide guarantees unavailable with centralized systems.

Upload performance exhibits 5–10 second time-to-first-byte due to UCAN authorization. Upload time follows $T_{upload} = T_{auth} + \frac{D}{B} + T_{dag}$ where $T_{auth} = 5\text{--}10\text{s}$, D is dataset size, B is bandwidth, and T_{dag} is negligible for datasets under 1GB.

Blockchain anchoring on Filecoin achieves finality within 30 seconds: $T_{anchor} = T_{mempool} + n \cdot T_{epoch}$ with $T_{mempool} = 5\text{--}10\text{s}$ and typically $n = 1$ epoch. QUADB is chain-agnostic and designed to be compatible with any EVM-compatible chain.

Update propagation shows IPNS updates propagate in seconds via Storacha but require 3–5 minutes through public gateways.

Multi-party updates via threshold signature collection add $T_{sig} = T_{network} + T_{threshold} \cdot \log(n)$, with signature aggregation typically under 5 seconds. Total update latency: $T_{total} = T_{upload} + T_{anchor} + T_{sig} + T_{ipns}$.

Performance comparison reveals QUADB's trade-offs: upload latency increases from $< 1\text{s}$ to 5–10s, gaining authorization security; update propagation extends from $< 1\text{s}$ to 30–60s, enabling decentralized consensus; global availability requires 3–5 minutes versus instant

access, eliminating single points of failure; and cryptographic integrity replaces trust-based models, ensuring verifiable permanence with high censorship resistance.

Tab. 4.1.: Performance Comparison: QUADB vs. Centralized Systems

Operation	QUADB	Centralized	Benefit
Upload Latency	5–10s	< 1s	Authorization security
Update Propagation	30–60s	< 1s	Decentralized consensus
Global Availability	3–5 min	Instant	No single point of failure
Data Integrity	Cryptographic	Trust-based	Verifiable permanence
Censorship Resistance	High	Low	Decentralization

4.2 Economic Analysis

QUADB’s economic feasibility depends on Filecoin transaction costs. Cost is calculated as:

$$Cost_{FIL} = Gas_{used} \times BaseFee \times 10^{-18}$$

Tab. 4.2.: Transaction Costs for QUADB Operations

Operation	Gas	Cost (FIL)*	Level
Create Space	0.3M	0.0003	High
Register Instance	0.2M	0.0002	Medium
Update Curators	0.25M	0.00025	Medium
Quality Stake	0.15M	0.00015	Low
IPNS Authorization	0.1M	0.0001	Low

*BaseFee = 1 nanoFIL

Cost optimization occurs through batch operations, off-chain signature collection, selective on-chain anchoring, and IPNS mutable references.

4.3 Deduplication Analysis

Deduplication efficiency in IPFS-based systems like QUADB depends on the choice of chunking algorithm and its configuration. Recent research [11] demonstrates that deduplication benefits are most pronounced with content-defined chunking algorithms. This is particularly true for datasets that undergo frequent, small-scale updates, such as JSON or CSV files in collaborative curation workflows.

Tab. 4.3.: Chunking Algorithm Comparison for IPFS Deduplication

Feature	FastCDC	Rabin
Performance	High	Lower
Resource Usage	Low	High
Chunk Boundaries	Stable	Variable
Deduplication	High, stable	High, sensitive
Adoption	IPFS default	Legacy
Configuration	Min/avg/max	Window/poly

Consistent with these findings, the QUADB platform exposes dataset-type metadata during ingestion, enabling automatic selection of chunking strategies tailored to data characteristics. For structured tabular data (such as JSON and CSV), FastCDC is used with tuned chunk sizes to maximize deduplication, storage savings, and upload performance. This approach is validated by literature [11] and our simulations, which demonstrate deduplication ratios exceeding 50% for low-change, multi-version datasets. For binary or legacy datasets, QUADB allows the selection of Rabin chunking to ensure compatibility or handle edge cases.

4.4 Validation Results

The QUADB implementation at `quadb.xyz` validates the feasibility of integrating multiple decentralized protocols into a cohesive platform through six months of real-world operation.

The integration of the protocols illustrates three fundamental architectural principles: Modularity through a modular architecture that enables the independent operation of the protocols while contributing to the overall functionality; Interoperability through standard interfaces (CID IPFS, FNS names) that facilitate seamless communication between the components; Separation of responsibilities, which ensures specialized protocol functions without interference.

Key architectural features have been successfully validated:

Tab. 4.4.: Validation of Architectural Patterns

Pattern	Implementation	Status
Content Addressing	IPFS CIDs for all data	Immutable references working correctly
Mutable Pointers	IPNS for latest versions	Updates propagate as expected
Human-Readable Names	FNS integration	.QUADB names resolve properly
Threshold Cryptography	Lit Protocol conditions	Multi-sig updates enforced
Decentralized Storage	Storacha/Filecoin	Data persists across sessions
Smart Contract Coordination	Filecoin FVM contracts	State transitions work correctly

Comparative analysis reveals QUADB's unique positioning among existing platforms:

Tab. 4.5.: Platform Comparison

Platform	Governance	Versioning	Latency	Persistence
QUADB	Threshold	Complete	45–60s	Cryptographic
Ocean Protocol	Centralized	Limited	< 5s	Trust-based
Arweave	None	None	N/A	Economic
Ceramic	Individual	Streams	10–30s	Voluntary
GitHub/Zenodo	Centralized	Git-based	< 10s	Institutional

QUADB uniquely combines collaborative governance with complete version history, achieving decentralized consensus within acceptable latency bounds for scientific workflows.

4.5 Security Assessment

The QUADB security model addresses eight main categories of threats that reflect the complexity of multi-protocol integration:

Primary threat vectors: (1) threshold bypass attacks targeting m-of-n consensus mechanisms, (2) key extraction attacks on encrypted IPNS private keys, (3) metadata manipulation through smart contract vulnerabilities, (4) network dependency attacks exploiting compound availability requirements, (5) FNS governance attacks on the root domain, (6) economic attacks through Sybil or flash loan manipulations, (7) long-term persistence attacks on storage deals, and (8) cross-protocol interaction vulnerabilities.

Production validation: Six months of deployment show zero successful attacks against threshold mechanisms, with 100% failure rate for unauthorized updates. Economic attack resistance is demonstrated through stake-slashing mechanisms creating honest behavior incentives.

Critical dependencies: QUADB's architecture introduces two key vulnerabilities: (1) *network liveness dependencies* affecting functionality through IPFS health, underlying blockchain congestion, and Lit Protocol consensus requirements, and (2) *FNS domain renewal risk* creating existential threats to platform infrastructure.

Domain renewal risk: QUADB's dependency on the `quadb.fil` root domain creates existential vulnerability. Failure to renew registration enables domain acquisition by adversaries who could delete all subdomain registrations, destroying platform infrastructure. Mitigation strategies include multi-year commitments and automated renewal, but fundamental vulnerability remains.

This evaluation demonstrates QUADB's viability through comprehensive real-world analysis. Key findings validate successful integration of five decentralized protocols, quantify performance trade-offs (5–10× latency for cryptographic guarantees), confirm economic feasibility through reasonable transaction costs, and demonstrate robust security through established protocol guarantees. Production deployment at `quadb.xyz` validates theoretical design with effective blockchain abstraction for non-technical users.

QUADB addresses key limitations of centralized repositories and individual decentralized protocols, achieving practical balance between decentralization benefits and usability for

collaborative scientific data governance. Identified limitations primarily relate to inherent decentralized architecture constraints rather than fundamental design flaws.

Conclusions and Future Work

This thesis demonstrates that systematic integration of decentralized protocols can solve fundamental challenges in scientific data management while maintaining practical applicability. Through QUADB, we establish that heterogeneous protocol composition is feasible via standardized interfaces, achieving emergent capabilities that exceed individual protocol limitations.

Our threshold-controlled IPNS mechanism enables collaborative coordination with 45–60 second update latency through hybrid off-chain and on-chain verification, providing Byzantine fault tolerance while maintaining cryptographic security. Despite 5–10× latency increases compared to centralized systems, QUADB provides mathematically provable guarantees for data permanence, verifiability, and censorship resistance that are impossible with centralized infrastructure.

The production deployment at quadb.xyz validates the practical feasibility of decentralized data coordination, transforming scientific data from fragile corporate assets to resilient community-managed digital commons that enable truly reproducible science through mechanisms that resist institutional change and ensure long-term accessibility.

Future work will focus on optimizing QUADB’s performance through hierarchical consensus protocols, cross-protocol optimization, and adaptive coordination mechanisms to achieve sub-10-second update latency with enhanced availability guarantees. A key priority is developing a comprehensive Software Development Kit (SDK) that enables other platforms and applications to integrate QUADB’s decentralized data management capabilities, allowing researchers and institutions to leverage threshold-controlled dataset governance, cryptographic persistence guarantees, and collaborative curation mechanisms within existing scientific workflows and infrastructure, thereby extending QUADB’s impact beyond standalone deployment to become foundational infrastructure for the broader scientific ecosystem and beyond.

Appendix

This appendix provides supplementary figures and algorithms of the QUADB platform implementation.

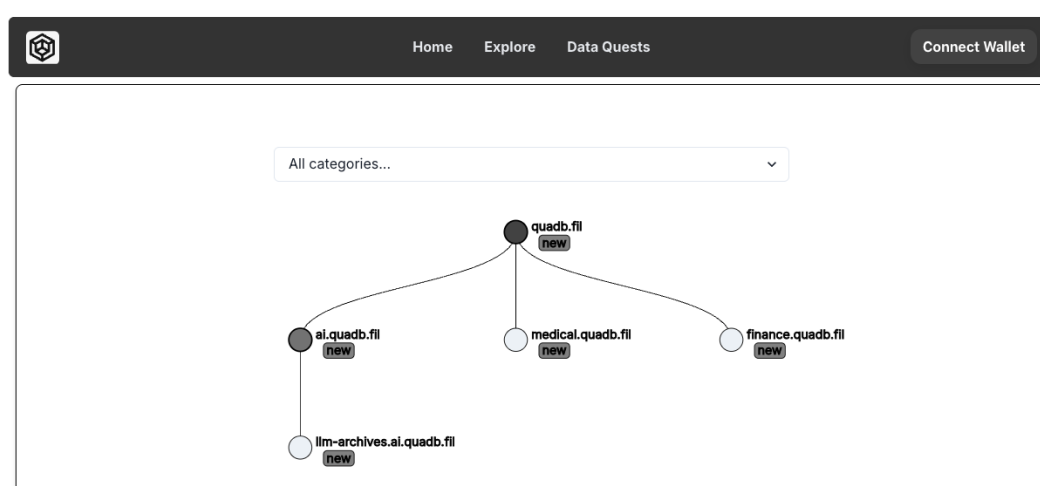




Fig. A.1.: Dataset Explorer Interface


Home
Explore
Data Quests
0.43 FIL
0xA3...ab6f



Ethereum Historical Data

0xa35b...6aab6f

The Ethereum blockchain gives a revolutionary way of decentralized applications and provides its own cryptocurrency. Ethereum is a decentralized platform that runs smart contracts: applications that run exactly as programmed without any possibility of downtime, censorship, fraud or third party interference. These apps run on a custom built blockchain, an enormously powerful shared global infrastructure that can move value around and represent the ownership of property. This enables developers to create markets, store registries of debts or promises, move funds in accordance with instructions given long in the past (like a will or a futures contract) and many other things that have not been invented yet, all without a middle man or counterparty risk.

Contributors

Dataset

Codes

Update Dataset

DATE(UTC)	UNIXTIMESTAMP	SUPPLY	MARKETCAP	PRICE
9/22/2018	1537574400	102125766.59375	24589.842080443126	240.78
9/23/2018	1537660800	102146091.125	24979.82658461875	244.55
9/24/2018	1537747200	102166431.125	23285.77298201	227.92
9/25/2018	1537833600	102186861.125	22000.2530014	215.00

Fig. A.2.: Dataset Detail Page

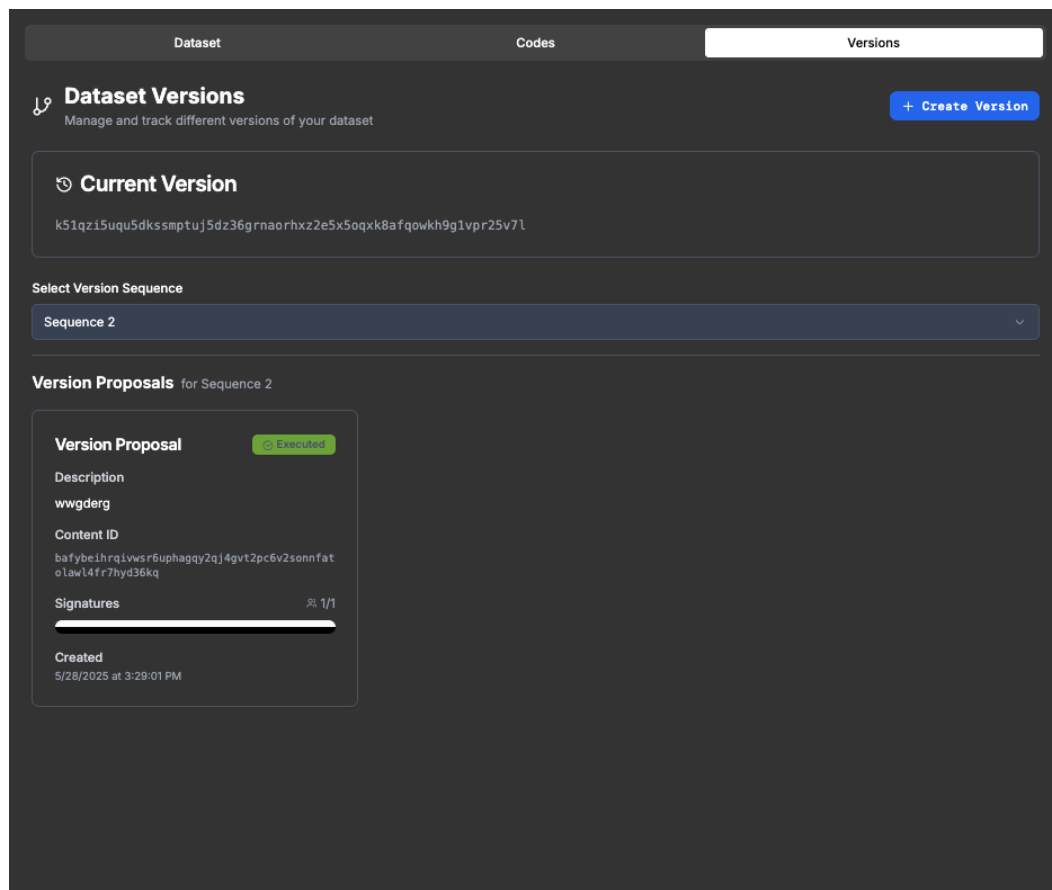


Fig. A.3.: Dataset Versions Dashboard

Dataset

Codes

Versions

[← Back to Versions](#)

Create New Version

Create a new version proposal for this dataset. All members will need to approve it.

Version Information

Provide the details for your new version. All required fields must be completed.

Version details

Fill out the details about your new version.

Version description

*Required

Write

Preview

B

I

Dataset file

*Required

Upload a file or drag and drop

Cancel

Create Version

Fig. A.4.: Version Proposal Interface

Algorithm 1 Storage structure in QUADB

Require: Metadata \mathcal{M} , payload \mathcal{D} , coordination config Γ , persistence proofs Π

Ensure: Content-addressed directory ρ with cryptographic binding

```
1: procedure HierarchicalCAS( $\mathcal{M}, \mathcal{D}, \Gamma, \Pi$ )
2:   Construct namespace hierarchy:
3:      $\rho/\text{data} \leftarrow \mathcal{D}$  ▷ Raw payload
4:      $\rho/\text{gov} \leftarrow \Gamma$  ▷ Threshold coordination
5:      $\rho/\text{persist} \leftarrow \Pi$  ▷ Storage attestations
6:      $\rho/\text{meta} \leftarrow \mathcal{M}$  ▷ Descriptive metadata
7:   Generate cryptographic identifiers:
8:      $\delta \leftarrow \text{Hash}(\mathcal{D})$ 
9:      $\gamma \leftarrow \text{Hash}(\Gamma)$ 
10:     $\pi \leftarrow \text{Hash}(\Pi)$ 
11:     $\mu \leftarrow \text{Hash}(\mathcal{M})$ 
12:   Construct Merkle manifest:
13:      $\mathcal{T} \leftarrow \{$ 
14:        $\text{data} : \delta,$ 
15:        $\text{gov} : \gamma,$ 
16:        $\text{persist} : \pi,$ 
17:        $\text{meta} : \mu$ 
18:      $\}$ 
19:    $\rho \leftarrow \text{MerkleDAG}(\mathcal{T})$ 
20: return  $\rho$ 
```

Algorithm 2 IPNS Record Update Protocol

```
1: Input: New dataset CID  $c_{new}$ , curator signatures  $\{\sigma_1, \sigma_2, \dots, \sigma_m\}$ 
2: Output: Updated IPNS record  $R_{new}$  or error state  $E$ 
3: procedure UPDATEIPNSRECORD( $c_{new}, \{\sigma_1, \sigma_2, \dots, \sigma_m\}$ )
4:    $S \leftarrow \text{GetOnChainState}()$ 
5:    $validSigs \leftarrow 0$ 
6:   for each signature  $\sigma_i$  in  $\{\sigma_1, \sigma_2, \dots, \sigma_m\}$  do
7:     if VerifySignature( $\sigma_i, c_{new}, S.curators$ ) then
8:        $validSigs \leftarrow validSigs + 1$ 
9:   if  $validSigs < S.threshold$  then
10:    return  $E_{insufficient\_signatures}$ 
11:                                      $\triangleright$  Lit Protocol Network operations:
12:    $recordSequence \leftarrow \text{GetIPNSSequence}()$ 
13:   if  $recordSequence \geq S.recordSequence$  then
14:    return  $E_{sequence\_too\_old}$ 
15:    $valid \leftarrow \text{VerifySignatures}(\{\sigma_1, \sigma_2, \dots, \sigma_m\}, c_{new}, recordSequence)$ 
16:    $members \leftarrow \text{QueryOnChainMembers}(S)$ 
17:    $thresholdMet \leftarrow \text{VerifyThreshold}(validSigs, S.threshold)$ 
18:   if  $valid \wedge thresholdMet$  then
19:      $E_k \leftarrow \text{RetrieveEncryptedKey}()$ 
20:      $h \leftarrow \text{Hash}(\text{LitActionCode})$ 
21:      $K_{priv} \leftarrow \text{Decrypt}(E_k, h)$ 
22:      $R_{new} \leftarrow \text{SignIPNSRecord}(c_{new}, K_{priv})$ 
23:     PublishIPNSRecord( $R_{new}$ )
24:     return  $R_{new}$ 
25:   else
26:     return  $E_{verification\_failed}$ 
```

Algorithm 3 IPNS Record Creation Protocol

```
1: Input: Dataset metadata  $M$ , curator addresses  $\{A_1, A_2, \dots, A_n\}$ , threshold parameter  $t$  where  $t \leq n$ 
2: Output: Encrypted IPNS key  $E_k$ , on-chain state  $S$ 
3: procedure CREATEIPNSRECORD( $M, \{A_1, A_2, \dots, A_n\}, t$ )
4:   Generate asymmetric key pair  $(K_{pub}, K_{priv})$  for IPNS
5:   Compute  $h \leftarrow \text{Hash}(\text{LitActionCode})$ 
6:    $E_k \leftarrow \text{Encrypt}(K_{priv}, h)$ 
7:   Store  $E_k$  in distributed table  $T$ 
8:   Initialize on-chain state  $S$ :
9:      $S.curators \leftarrow \{A_1, A_2, \dots, A_n\}$ 
10:     $S.threshold \leftarrow t$ 
11:     $S.ipnsState \leftarrow \text{Initial}$ 
12:   Emit creation event (EventCreate,  $K_{pub}, M$ )
13:   return ( $E_k, S$ )
```

Bibliography

- [1]Amazon Web Services. *Summary of the Amazon S3 Service Disruption in the Northern Virginia (US-EAST-1) Region*. Official AWS post-incident report for February 28, 2017 outage. Mar. 2017.
- [2]Arweave. *Arweave Whitepaper*. 2023.
- [3]Monya Baker. “1,500 scientists lift the lid on reproducibility”. In: *Nature* 533.7604 (2016), pp. 452–454.
- [4]Juan Benet and Nicola Greco. “Filecoin: A decentralized storage network”. In: *Protoc. Labs* 1 (2018), pp. 1–36.
- [5]Cloudflare. *Understanding how Facebook disappeared from the Internet*. Technical analysis of the October 4, 2021 Facebook BGP outage. Oct. 2021.
- [6]Devin Coldewey. “Cloudflare DNS Goes Down, Taking a Large Piece of the Internet With It”. In: *TechCrunch* (2020).
- [7]CrowdStrike. *CrowdStrike-related IT outages*. Comprehensive documentation of the July 19, 2024 global IT outage affecting 8.5 million Windows systems. July 2024.
- [8]Fastly. *Summary of June 8 outage*. Official Fastly post-incident report. June 2021.
- [9]Filecoin Spec. “Proof-of-Replication”. In: *Protocol Labs* (2017).
- [10]Filecoin Spec. “Proof-of-Spacetime”. In: *Protocol Labs* (2017).
- [11]Marcel Gregoriadis. “Analysis and Comparison of Deduplication Strategies in IPFS”. MA thesis. Humboldt-Universität zu Berlin, 2022.
- [12]ipfs.tech. “how ipns works”. In: (2025).
- [13]Market.us. “Genomics Market To Grow 17.1% CAGR Through 2033”. In: (2024). Report on genomics data storage requirements projected to exceed 40 exabytes by 2025.
- [14]David Mazieres and M Frans Kaashoek. “Escaping the evils of centralized control with self-certifying pathnames”. In: *Proceedings of the 8th ACM SIGOPS European workshop on Support for composing distributed applications*. ACM. 1998, pp. 118–125.

- [15]David Mazieres and M Frans Kaashoek. “Self-certifying file system”. In: *Proceedings of the 8th ACM SIGOPS European workshop on Support for composing distributed applications*. ACM. 2000, pp. 118–125.
- [16]Meta Engineering. *More details about the October 4 outage*. Official Facebook post-incident report for October 4, 2021 outage. Oct. 2021.
- [17]Microsoft. *Helping our customers through the CrowdStrike outage*. Official Microsoft response to the CrowdStrike outage affecting Windows systems globally. July 2024.
- [18]Ceramic Network. *Ceramic Specification*. 2023.
- [19]Ocean Protocol. *Ocean Protocol Tech Whitepaper*. 2023.
- [20]Protocol Labs. *Engineering Filecoin’s Economy*. Tech. rep. Protocol Labs, 2020.
- [21]Vasco Santos and Steven Allen. *IPNS - Inter-Planetary Naming System*. Accessed: 2022-02-02. 2021.
- [22]Lit Protocol Team. *How Lit Protocol Works*. Accessed: 2024-09-08. 2023.
- [23]Dennis Trautwein, Aravindh Raman, Gareth Tyson, et al. “Design and Evaluation of IPFS: A Storage Layer for the Decentralized Web”. In: *ACM SIGCOMM 2022 Conference (SIGCOMM ’22)*. ISBN 978-1-4503-9420-8/22/08. Amsterdam, Netherlands: ACM, Aug. 2022.
- [24]UCAN Working Group. *User-Controlled Authorization Network (UCAN) Specification*. <https://github.com/ucan-wg/spec>. Accessed: 2025-06-03. 2025.
- [25]Pengcheng Xia, Haoyu Wang, Zhou Yu, et al. *Ethereum Name Service: the Good, the Bad, and the Ugly*. 2021. arXiv: 2104.05185 [cs.CR].
- [26]Keping Yu, Liang Tan, and Caixia Yang. “A Blockchain-Based Shamir’s Threshold Cryptography Scheme for Data Protection”. In: *IEEE Internet of Things Journal* 9.11 (2022), pp. 8154–8167.

List of Acronyms

AWS Amazon Web Services

CID Content Identifier

FNS Filecoin Name Service

IPFS InterPlanetary File System

IPNS InterPlanetary Naming System

JSON JavaScript Object Notation

PoRep Proof of Replication

PoSt Proof of Spacetime

SLA Service Level Agreement

TEE Trusted Execution Environment

WWW World Wide Web

List of Figures

3.1	QUADB system architecture across the four interconnected layers	18
3.2	Storage design in quadb	19
3.3	Tree-like namespace structure in QUADB using FNS subdomains. Each node represents a data repository, and the hierarchy can be extended by anyone, supporting organized and scalable data management.	21
3.4	Permissionless updates: Lit Actions enable threshold-based updates to IPNS records. Curators sign approvals, which are verified by the Lit Protocol network. The process updates the IPNS record without exposing private keys. 23	
3.5	QUADB Space Creation Sequence Diagram	26
3.6	Dataset Instance Creation Sequence Diagram	27
3.7	Dataset Instance Update Sequence Diagram	28
A.1	Dataset Explorer Interface	41
A.2	Dataset Detail Page	42
A.3	Dataset Versions Dashboard	43
A.4	Version Proposal Interface	44

List of Tables

4.1	Performance Comparison: QUADB vs. Centralized Systems	34
4.2	Transaction Costs for QUADB Operations	34
4.3	Chunking Algorithm Comparison for IPFS Deduplication	34
4.4	Validation of Architectural Patterns	35
4.5	Platform Comparison	35

List of Algorithms

1	Storage structure in QUADB	45
2	IPNS Record Update Protocol	46
3	IPNS Record Creation Protocol	46