Project Plan Report

CS 6650: Building Scalable Distributed Systems

Project Title: Blockchain-Backed Distributed File Storage

Team Members: Samyak Shah, Maheep Parekh

Overview

- We plan to design a small-scale distributed file storage system where files are divided into chunks and distributed across multiple storage nodes.
- A lightweight blockchain ledger maintains the metadata of stored files, recording root hashes (CIDs), uploader information, and storage proofs.
- The project demonstrates how blockchain ensures integrity and verifiability in decentralized storage.

Goals

- Implement a DHT-based file storage system that distributes and retrieves file chunks efficiently.
- Integrate a blockchain layer to record metadata and proofs of storage.
- Ensure end-to-end verification: every file retrieved matches its root hash.
- Demonstrate decentralized trust - no central server needed to verify data authenticity.

Detailed System Design

- **Architecture Layers:**
  - **Storage Layer (DHT)** – Handles chunking, hashing, and peer-to-peer chunk storage.
  - **Blockchain Layer** – Stores metadata transactions (`FileAdded`, `ProofOfStorage`) and validates proofs.
  - **Client Interface** – CLI-based client to upload, retrieve, and verify files.
- **Workflow:**
  1. User uploads a file; it is split into equal size, each hashed with SHA-256.
  2. A root hash (CID) is computed and stored on the blockchain.
  3. Chunks are distributed among storage nodes via DHT lookups.
  4. During retrieval, the client verifies chunk integrity using the recorded CID.
- **Verification:** Blockchain smart contract validates proofs and logs verified nodes.

Assumptions

- Nodes may fail independently; network partitions are temporary.
- All nodes are connected.
- File sizes will be moderate.
- Each node can store a few chunks; at least two replicas per chunk.
- No change-of-ownership or cryptocurrency incentives in this version.

Modules

- **Client Module** – Uploads/retrieves files, computes CIDs, communicates with blockchain.
- **DHT Module** – Maintains key-value mappings (CID $\rightarrow$ node list).
- **Storage Node Module** – Stores chunks locally and responds to retrieval requests.
- **Blockchain Module** – Records `FileAdded` and `ProofOfStorage` transactions.
- **Verification Module** – Cross-checks hashes between storage and blockchain records.

How We Would Evaluate the System

Functional Testing

Each key feature (upload, download, verify, update) will be tested individually to ensure correctness.

Integrity Verification

We will intentionally modify a file chunk and verify that the system detects the inconsistency using blockchain metadata.

Fault Tolerance

We will simulate node failures by shutting down a node process and confirm that replication or fallback mechanisms allow successful file retrieval.

Performance Evaluation

Measure average DHT lookup latency and blockchain update propagation time to assess scalability and efficiency.

Cluster Deployment

All nodes and clients will be deployed on the Khoury Linux cluster using SSH-hosted servers. Each server will simulate a distinct node participating in the DHT network and blockchain replication.

Step-by-Step Plan and Timeline (Tentative)

- **Week 1:** Initial setup, define file formats, and implement basic client-server communication.
- **Week 2:** Implement DHT-based lookup and node registration.
- **Week 3:** Integrate blockchain ledger for file metadata and version tracking.
- **Week 4:** Implement verification logic and test node failure recovery.
- **Week 5:** Perform full evaluation on the cluster, collect results, and prepare final report.

Summary

This project integrates the scalability of DHTs with the trust and immutability of blockchain to create a verifiable distributed file storage system. The design will remain modular and flexible to accommodate future refinements as the implementation evolves. The project will demonstrate distributed coordination, integrity verification, using a real multi-node setup on Khoury Linux SSH servers.
