Project Proposal: Decentralized Intranet Resource-Sharing Platform

Team 1
Name: Rudra Ayachit and Shreyas Guggilla
Date: 25th September

Project Overview

Modern organizations often under-utilize computing and storage resources across departments and labs. This project proposes the development of a private peer-to-peer (P2P) infrastructure that allows nodes on an intranet to discover each other, share storage, and execute compute jobs without a central server. The system will enable secure, fault-tolerant, and efficient use of existing hardware while remaining fully isolated from the public internet.

Objectives

• Build a private P2P overlay network for automatic node discovery and secure communication.
• Implement distributed storage so that files added on any node are available to all peers.
• Provide a compute-sharing layer that schedules and executes containerised jobs across available machines.
• Ensure data confidentiality and node authentication through strong cryptography.
• Deliver a scalable proof-of-concept (PoC) that can expand to additional nodes or future incentive models.

Scope

Intranet only: All traffic remains inside the organization's private network.
Resources shared: Disk storage (for file hosting and retrieval) and CPU/GPU cycles for lightweight batch jobs.
Out of scope: Public blockchain integration, complex billing mechanisms, wide-area deployment.

System Architecture

1. P2P Networking Daemon – Built in Go using libp2p for peer discovery (mDNS) and encrypted communication.
2. Distributed Storage – IPFS private network with a shared swarm.key for content-addressed file sharing.
3. Compute Runner – Docker Engine API to launch sandboxed jobs on volunteer nodes, with a simple scheduler integrated into the daemon.
4. Security & Authentication – Mutual TLS using an internal Certificate Authority and optional role-based access control.

Technology Stack

Language: Go
P2P Networking: libp2p
Storage: IPFS (private mode)
Compute: Docker, Docker Engine API

Security: TLS, internal PKI
Monitoring: Prometheus, Grafana (optional)
Deployment: Linux servers/VMs, Docker Compose

Implementation Plan & Timeline

Phase 1 – Setup & Foundations (Week 1–2): Environment setup, certificate authority, Go/libp2p peer discovery.
Phase 2 – Storage Integration (Week 3): Private IPFS network, CLI for put/get commands.
Phase 3 – Compute Runner (Week 4): Docker job execution via libp2p messaging.
Phase 4 – Security & Monitoring (Week 5): Mutual TLS, Prometheus metrics, dashboard.
Phase 5 – Testing & Documentation (Week 6): End-to-end PoC demo, user guide, final report.

Expected Outcomes

• A working multi-node PoC demonstrating file sharing and remote job execution within a closed intranet.
• Source code and deployment scripts for easy replication.
• Documentation of architecture, APIs, and security model.

Potential Extensions

• Incentive/credit system using a private blockchain or internal token.
• Advanced scheduling with resource load balancing.
• Integration with Kubernetes for large-scale container orchestration.
• Edge-network features like bandwidth sharing or internal CDN.

Resources & Requirements

Hardware: 3–5 Linux servers or virtual machines (2 vCPU, 4 GB RAM each).
Software: Go compiler, Docker, IPFS, Git.
Team Skills: Go programming, Linux administration, distributed systems, and cryptography/TLS configuration.

Impact

The project will enable efficient use of under-utilized organizational hardware, reduce dependency on centralized servers, and provide a secure foundation for future distributed applications such as internal data lakes, analytics pipelines, or AI model training.