Abstract:

Distributed Filesystem architectures have enabled usage of commodity hardware to store data on a large scale with maximum consistency and availability. Blockchain makes it possible to store information which can never be tampered with and allows for incentivization of a traditional decentralized storage system. The goal of this project is to implement a decentralized filesystem which leverages blockchain to keep a record of all the transactions. Unlike a conventional filesystem viz. GFS or HDFS which uses designated servers owned by their organization to store the data and are governed by a Master server this project aims to remove a single point of failure and makes use of participating user machines to store data. The first task is to implement a distributed hash table which can be used to evenly store data on multiple machines and efficiently track, maintain, and persist the data. Secondly the data needs to be encrypted and easily and readily accessible to a user ensuring its soundness. Finally using smart contracts enables incentives and increases integrity and reliability of storing data by recording all the transactions. Our project will function autonomously by making use of multiple participant machines in the network to store data and monetize unused storage space. It will support basic functionalities of a filesystem which include creation, deletion, reading of a file along with storing it in a hierarchical folder like structure.

Keywords - Distributed Filesystem, Blockchain, Chord, DHT, Smart Contracts

CS297 Results

- Implemented a simple CHORD distributed hash table.
- Carried out a comparative study between existing decentralized storage systems which make use of blockchain namely IPFS, Storj, SWARM and SIA.
- Developed a basic structure of the filesystem which could be used along with a distributed hash table by partitioning the files and maintaining the integrity of data bits using Merkle trees.
- Created a simple gambling game using smart contracts on Ethereum blockchain to get a better understanding of Solidity and Smart contracts.

Proposed Schedule

Week 1: Jan 30 - Feb 6	First Week Meeting and Reviewing CS298 Proposal
Week 2: Feb 6 - Feb 13	Start implementing CHORD protocol
Week 3: Feb 13 - Feb 20	Continue with the CHORD implementation
Week 4: Feb 20 - Feb 27	Integrate CHORD into the filesystem structure

Week 5: Feb 27 - Mar 6	Work on the edge cases and introduce encryption of data bits
Week 6: Mar 6 – Mar 13	Ensure end to end working of file storage using CHORD with multiple devices.
Week 7: Mar 13 - Mar 20	Start working on the smart contract
Week 8: Mar 20 - Mar 27	Continue working on the smart contract along with web3 implementation
Week 9 and 10: Mar 27 - Apr 10	Integrate smart contract in the filesystem and ensure working of CHORD along with blockchain
Week 11: Apr 10 - Apr 17	End to end testing of the filesystem using blockchain
Week 12: Apr 17 - Apr 24	Result analysis and start working on the report
Week 13: Apr 24 - May 1	Continue to work on the report and get it reviewed by Chris
Week 14: May 1 - May 8	Finish report, start working on slides and get it reviewed by the project committee
Week 15: May 8 - May 15	Finish slides for the presentation
Week 16: May 15 - May 22	Wrap up all work

Key Deliverables:

- **Software**:
  - Successful storage of small (A few KB) and large (A few GB) files into the filesystem without having a single point of failure using a DHT (Preferably CHORD). Storage in folder structure and easy deletion as required.
  - Quick retrieval of files stored in the filesystem while ensuring the soundness of the data using a content ID which is generated during data storage in the filesystem.
  - Easy access to an Incentivization layer available for renting or providing storage space to any user using cryptocurrency
  - Extra layer of data integrity by making every user accountable for data stored on his machine by recording transactions on blockchain.
- **Report**
  - CS298 Report
  - CS298 Presentation

Innovations and Challenges

- Conventionally we store our data on commodity servers owned by organizations like Google, Amazon, Microsoft etc., but our filesystem will allow any user with a personal device to earn some extra bucks by renting his unused storage space to other users using our system.
- Fully autonomous filesystem with no master service to govern the control or the data plane.
- It will be crucial to have error handling and covering majority of the edge cases while storing the data on multiple servers.
- It will be challenging to introduce replication and mapping of replicas in a self-governed CHORD protocol.
- It will be important to make sure that only a limited amount of gas is spent by each user while recording transactions on the blockchain.