Weekly Progress Log

Nick Tagliamonte

5/25/25

Task

Install IPFS, simulate a small IPFS cluster, add a large file and test retrieval time, and familiarize with IPFS commands.

Steps Taken

- Installed Kubo (go-ipfs) on the host machine via official binaries.
- Initialized a local IPFS node and verified installation.
- Wrote a `docker-compose.yml` file to spin up 10 IPFS containers (node1 to node10), each with its own ports and volumes.
- Manually bootstrapped each containerized IPFS node to the host node using the output from `ipfs id` and `ipfs bootstrap add`.
- Used basic IPFS CLI commands: `add`, `cat`, `pin rm`, and `repo gc`.
- Verified that once a file is cached by other nodes via `cat`, it persists even after unpinning and garbage collection from the original node.
- Created a shell script to:
  - Copy a test MP4 file to all nodes.
  - Add the file on each node, record the CID and timing.
  - Retrieve it from every other node, logging retrieval time.

Findings

- File add times varied significantly (typically 40–90 seconds).
- Retrieval times were consistently fast (usually < 0.01 seconds).
- Caching works well across nodes, though not persistent unless pinned.

5/26/25

Task

Understand how replication works in IPFS; begin exploring IPFS Cluster and Filecoin incentives.

Findings on IPFS Replication

- Files are pinned on the node where they're added.
- When accessed by another node, they are cached but not pinned.
- Garbage collection on unpinned nodes can delete those cached copies.
- Due to the DHT, nodes may serve from a cached neighbor if it's closer in routing terms.

IPFS Cluster

- Cluster allows coordinated pinning across multiple IPFS nodes.
- Each IPFS node runs an associated `ipfs-cluster-service` daemon.
- A cluster-wide command (`ipfs-cluster-ctl pin add <CID>`) will distribute pins per policy.

Next Steps

- Install and configure IPFS Cluster.
- Link cluster nodes to IPFS nodes 1–10.

5/28/25

Task

Begin exploring Bitcoin Core from *Mastering Bitcoin*, Chapter 3.

Actions

- Cloned the Bitcoin Core repo and checked out tag `v29.0`.
- Installed necessary packages on Fedora via `dnf`.
- Built the Bitcoin client using `cmake`.
- Ran `bitcoind`, downloaded full blockchain data (over 700 GiB).

5/29/25

Task

Explore Bitcoin RPC interface and code examples.

Work Done

- Modernized several examples from Chapter 3 using `AuthServiceProxy`.
- Tested `getblockchaininfo`, `getrawtransaction`, and others.
- Implemented example scripts to:
  - Decode transactions and compute output values.
  - Compute total block value.

5/30/25

Task

Experiment with private key formats, wallet commands, and address generation.

Notes

- `dumpprivkey` is deprecated for descriptor wallets (modern default).
- Rewrote Examples 4-3, 4-5, and 4-7 using current libraries:
  - `ecdsa`
  - `cryptos`
  - `hashlib`, `base58`, `os`
- Implemented manual base58 address encoding.
- Generated compressed and uncompressed public keys.

5/31/25

Open Questions and Reflections

- What is the connection between Bitcoin, IPFS, and Filecoin in distributed systems research?
- Is IPFS effective in small networks, or does it rely too much on the DHT scale?
- What are the critical architectural differences between Bitcoin, IPFS, and BitTorrent?
- Can Bitcoin node operation be lightweight via pruning?