Network File System

(I’m sorry)

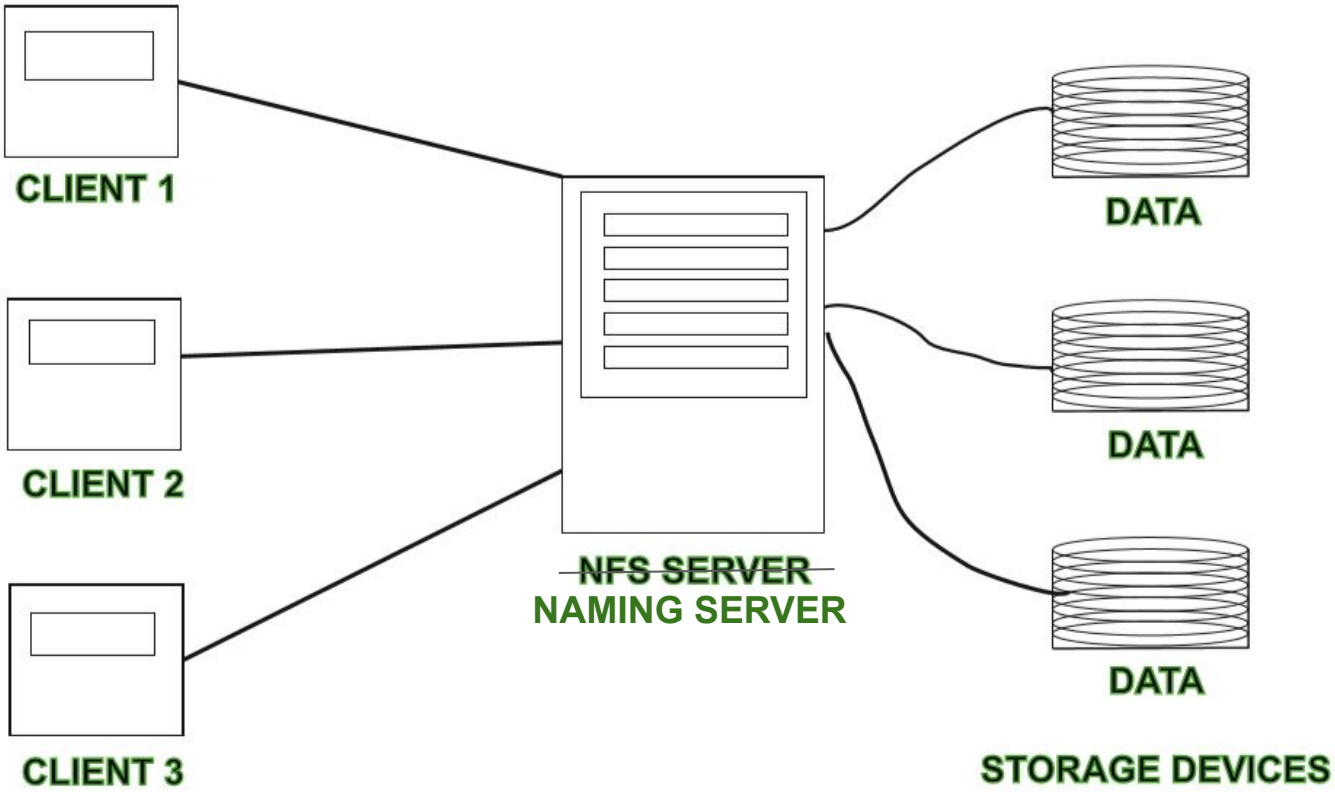
What’s included in this project?

~~Distributed Systems?~~

1. Socket programming (Briefly)
2. Threading
3. Concurrency

- OS Concepts (Scheduling, etc.)
- Data Structures and Algorithms
- API Calls?

NFS


CLIENT 1
CLIENT 2
CLIENT 3
NFS SERVER
NAMING SERVER
DATA
DATA
DATA
STORAGE DEVICES

The storage servers

- Store the data (duh)
- Expose the accessible paths to the NM
- Give relevant information such as ports, IP addresses, etc. to the NM on connection

Naming Server

- Can add new storage servers at any point in the execution (try to create a thread for adding new SS)
- Requests made to NM
  - Handled by client
    - Reading files
    - Writing files
    - Access information
  - Handled by NM
    - Creating files and directories
    - Deleting files and directories
    - Copying files and directories
  - **A few places require ACK; a few requirements are asynchronous.**

Naming Server

- I expect the previous parts to be relatively straightforward
- They account for ~50% of the assignment
- Let’s take a look at the other specifications

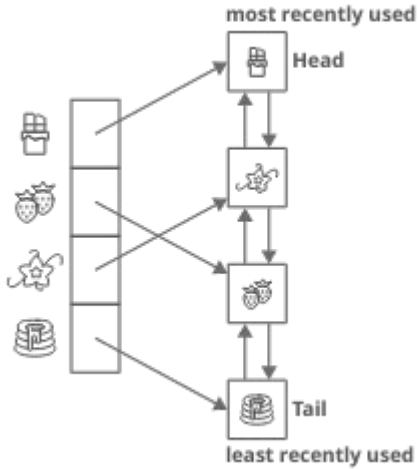
Multiple clients

Essentially your NM's *mantra* (Concurrent Client Access)

- File reading
- File Writing

Search

- Linear search?
  - 🤮
- No, do something better than *O(n)*
  - *Tries and hashmaps* (You can copy code from online)
- Caching

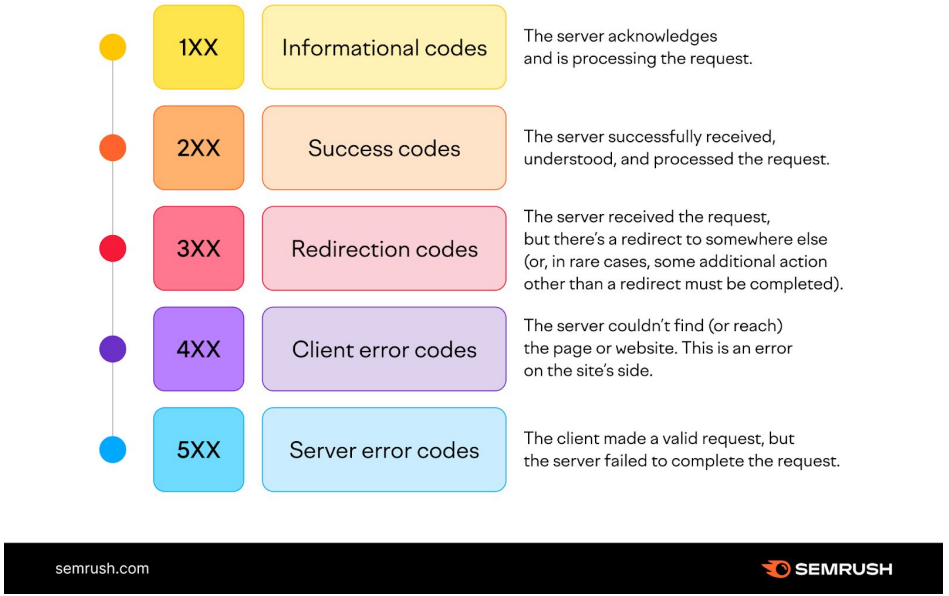


Redundancy and replication

- Easily the *most challenging* part of your assignment
- Failure detection
- Data redundancy and replication
  - Async
- SS Recovery
- Async duplication of write commands

Other stuff

- Error codes
  - Will make your own life easier
- Bookkeeping
  - Essentially debug statements



Motivation

- NFS is cool
- You get to build a proper *large* system (unlike C-shell)
- Learn a bunch of stuff
- If none of this worked
  - Think about what it can do for your resume :)

I asked Google for “Tech is cool” images

Some pointers

- Only POSIX Libraries
- If you’re picking up something from - cite the resource
- Prompts given to LLMs should be included in the code
- Modular code
- Divide work efficiently; plan
- Should have started yesterday

Thanks for coming

Especially when the other TAs (Swayam) were too busy to accompany me