Project 2 Guidelines

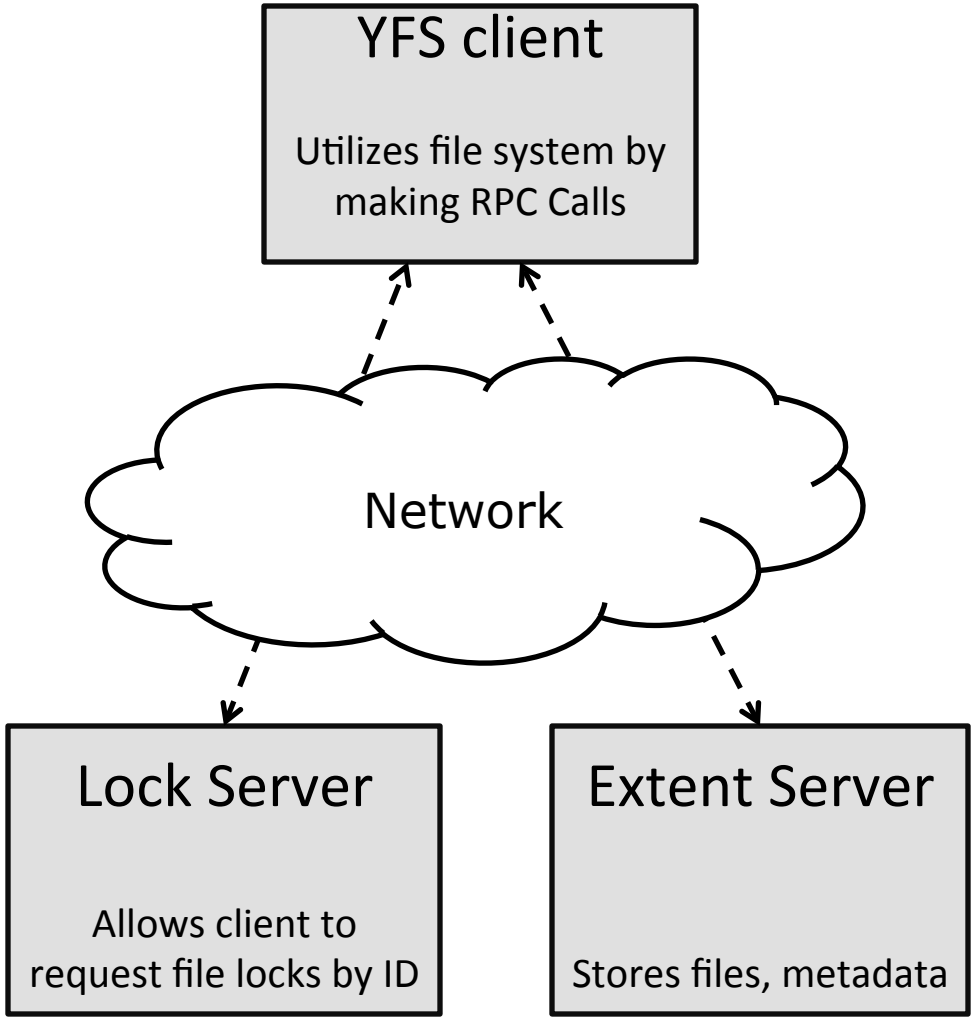
15-440 Spring 2011

YFS

- “Yet Another File System”/ “Your File System”

- Simple distributed file system
  - One extent server and one lock server
  - Entire state stored remotely
  - Logic for file system operations is actually in the client

YFS Communication



YFS Development

- Built incrementally
  - Lock Server
  - At-most-once RPC
  - File system operations
    - Client-side (FUSE) and server-side (Extent Server)
  - Caching of extents and locks

What is FUSE?

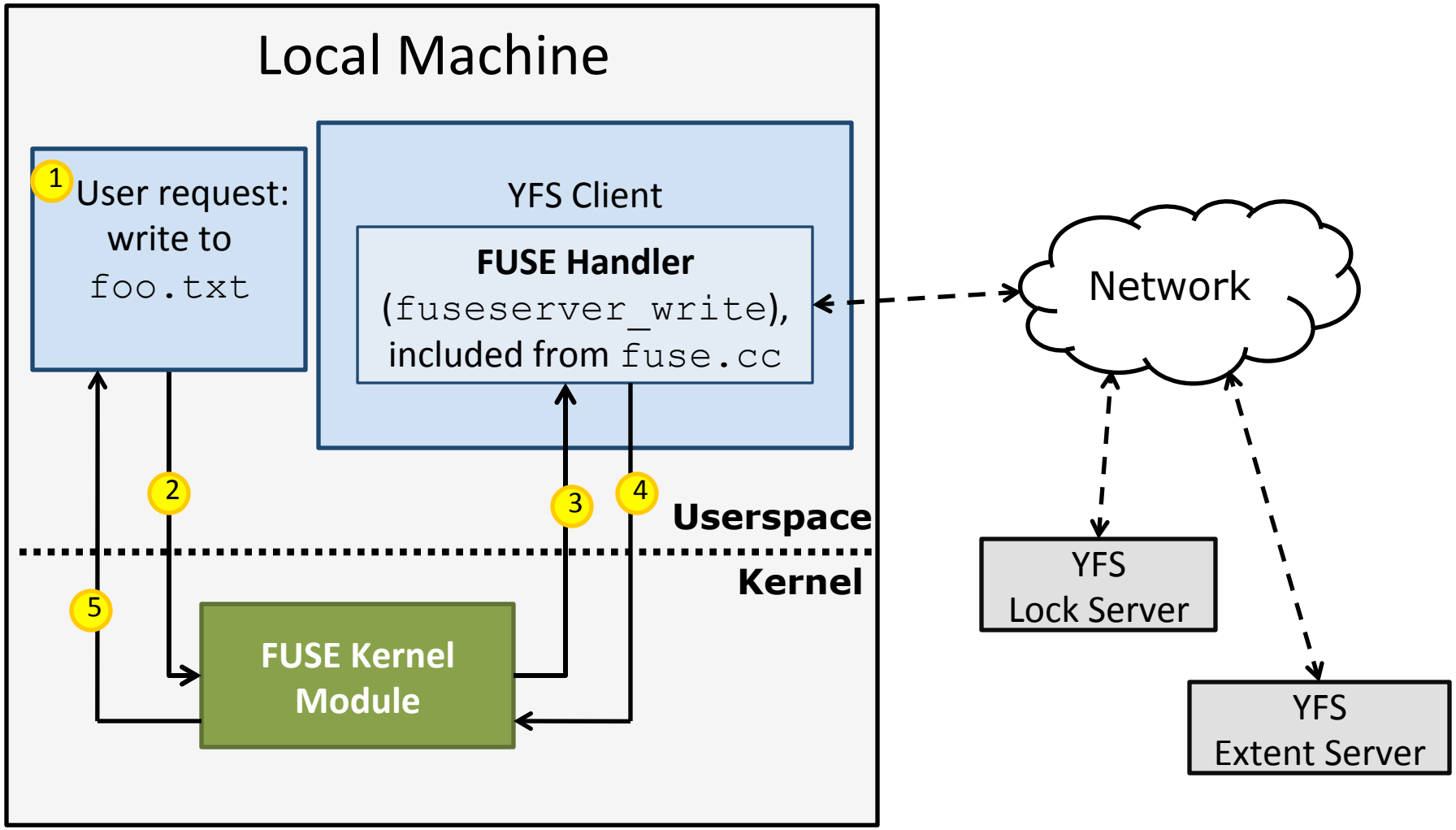
- **FUSE (Filesystem in Userspace)** is a kernel module that allows you to implement a file system in the user space

- We will use it in lab 2 to implement the file system logic without having to modify kernel code

FUSE and yfs

- You cannot read and write directly to the distributed file system like you would to a disk
  - A file "read" needs to contact the server to get the file, since it's stored remotely
- FUSE allows you to execute any instructions you'd like when a file system operation occurs
  - You will implement "handlers" to specify what to do when a user performs a file system operation
  - Hint: it should be contacting the servers!

YFS Execution Flow



YFS Execution Flow

1. The user requests to write to `foo.txt`

2. The **fuse kernel module** calls the appropriate **fuse handler**
   - in this case, it's the `fuseserver_write` handler in `fuse.cc`

3. Your (user-space) code is now in control!
   - Your code should communicate with the servers to carry out the write

4. The fuse handler communicates the status of the operation back to the kernel module

5. The status is reported back to the user

Development Environment

- FUSE is already configured in the Ubuntu VirtualBox image we provide
- `main()` starter code in `fuse.cc` does the low-level setup to make the file system accessible
- The YFS file system shows up on your desktop, just as a local disk would
  - Try running `./start.sh` to see this

Metadata

- **atime**
  - Last access time (read)
- **ctime**
  - Time of last "status" change (to file metadata)
  - Changes with `chmod, chgrp, chown`
  - Also updated when file contents change
- **mtime**
  - Time of last modification (of actual file data)
  - Should occur if writing > 0 bytes

Questions Seen So Far

- VirtualBox
  - Windows users, use version 3.2
- At-most-once RPC
  - Uses sliding window similar to TCP
  - Figure out the invariant the `rpcc` client follows
    - Use this to determine what you can "forget" on the server
  - `rpctest` may print "Connection refused" errors
- Make sure both partners work on key components
  - Locking, RPC, good to know for homeworks/exams