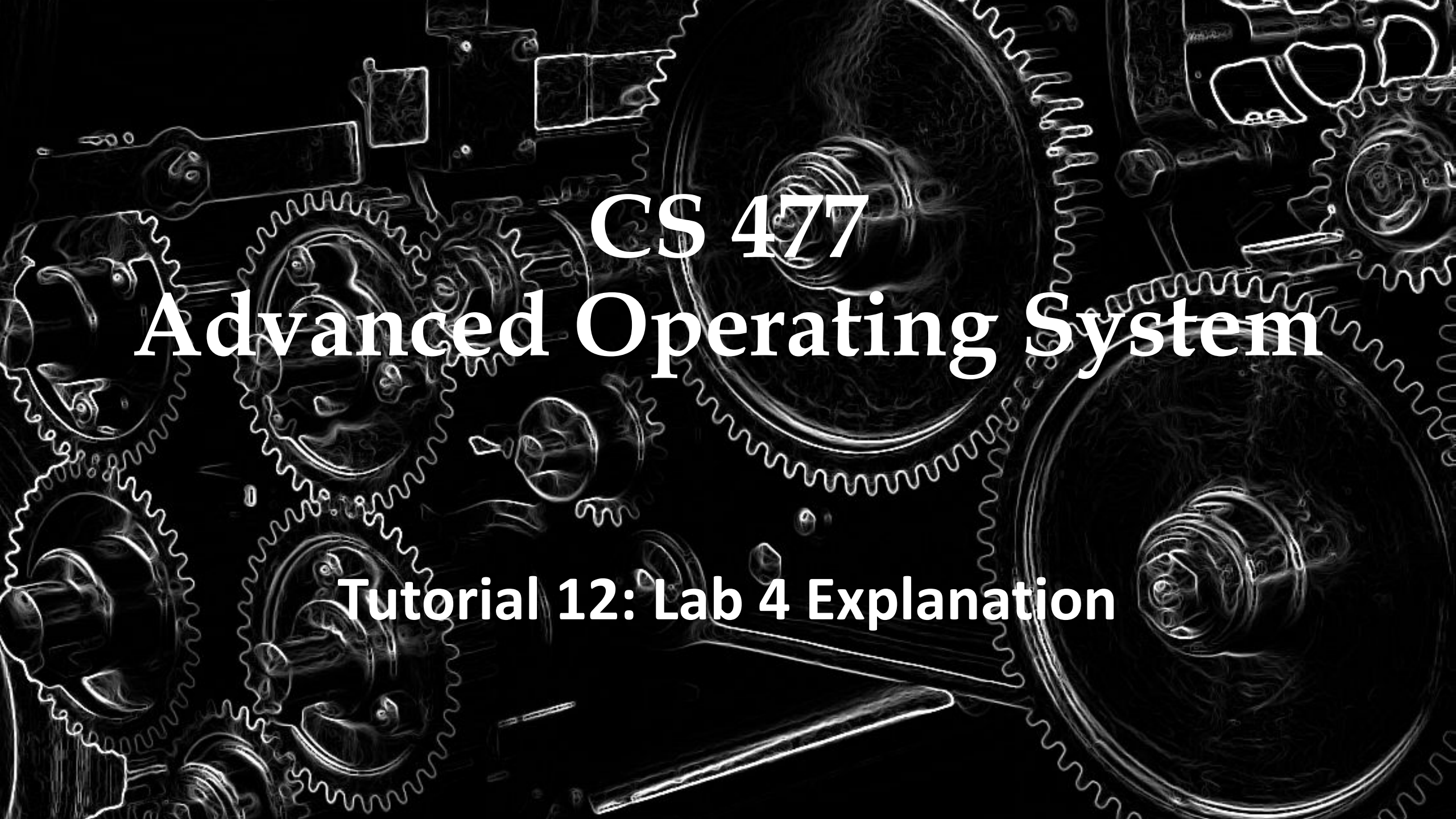


CS 477
Advanced Operating System

Tutorial 12: Lab 4 Explanation

Today's Tutorial Outline

- 15 mins: Slides
  - The "10000 foot view" of what each part of lab4 asks you
- 30 mins: Q&A
  - I'll write questions + attach doc w/ Q&A in Moodle after the session
- Interrupt me any time. Today is about clearing up details for you.

What can you do by the end of L4FS

- Format + mount a device (e.g. a USB)
- Use ls to check the contents of the device
- Create a new file in the device
- Read from/write to the device
- Enable crash consistent/concurrent FS

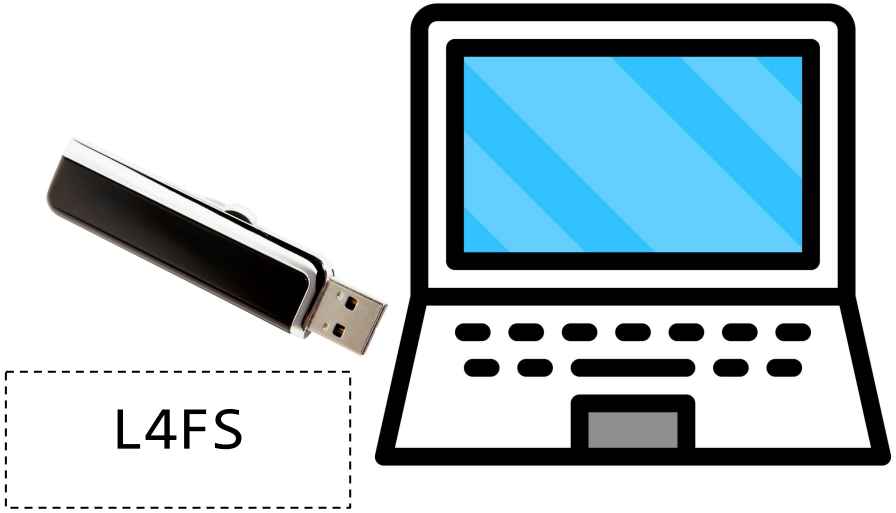
Part 1.2: *mkfs*

"Do you want to format this device"

`mkfs /dev/sda`

- We give you 2 variants
  - mkfs - clean install
  - mkfs_premade - w/ some files initialized
- **Your job: understand what mkfs initializes.**

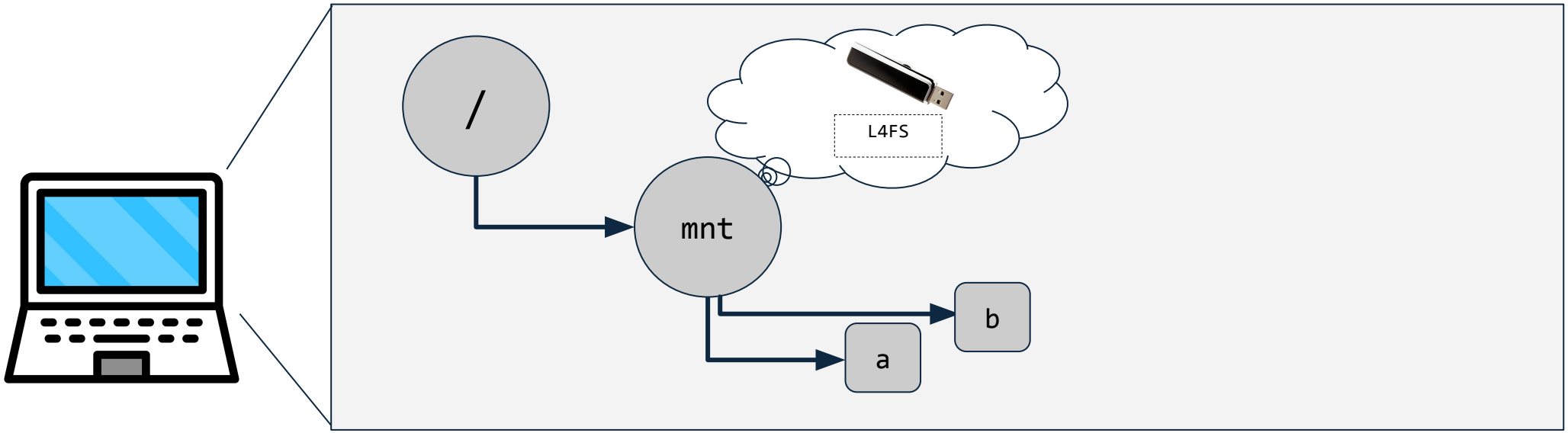
Part 1.3: *mount*




```
sudo mount /dev/sda -t l4fs /dev/sda /mnt
```

- We provide you skeleton that enables mount/unmount in the *l4fs* directory
- **Your job: explore how VFS registers a FS and mounts a new device**
  - **Answer questions scattered throughout l4fs kernel module**

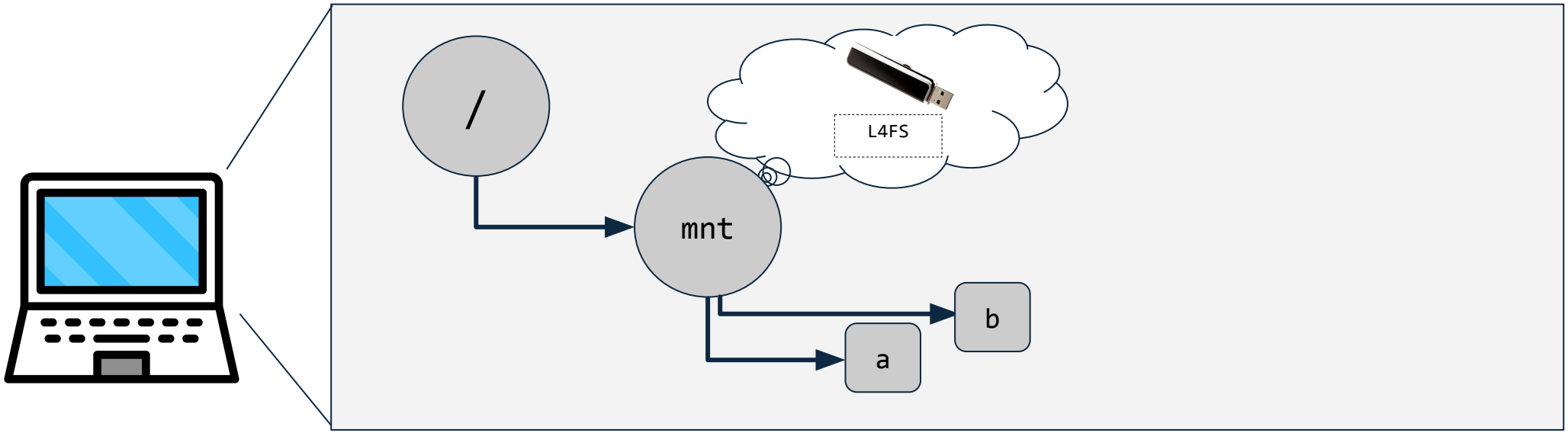
Part 2.1: *getdents64*




```
ls /mnt
```

- `(user)ls -> (syscall)getdents64 -> (vfs)iterate_dir -> (fs)iterate_shared`
- **Your job: implement iterate_shared**

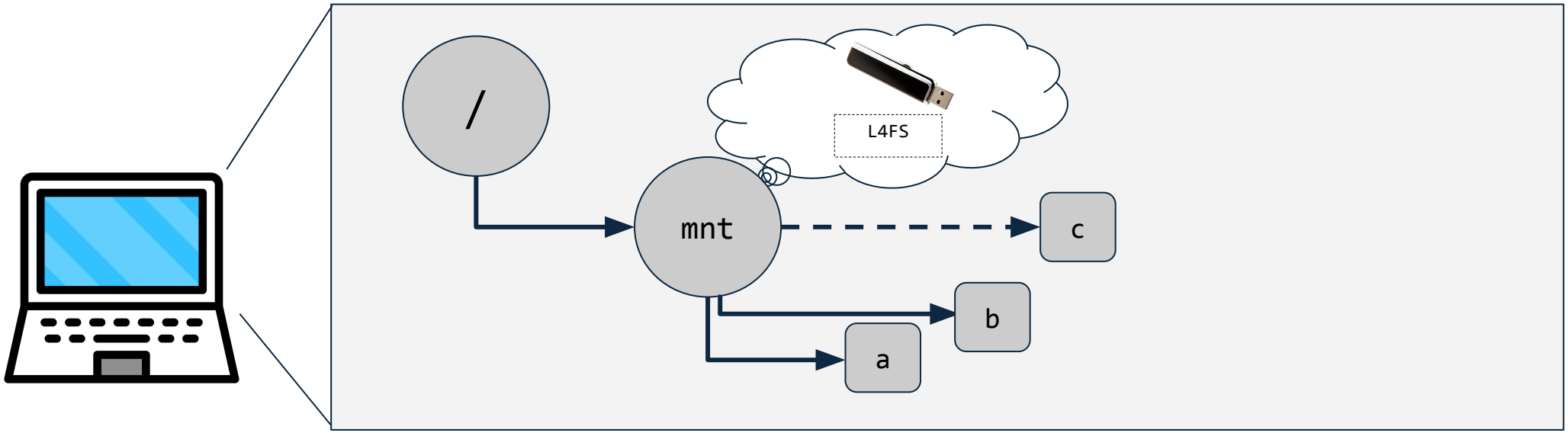
Part 2.2: *lookup*




```
open /mnt/a
```

- (syscall)open -> (vfs)do_filp_open -> (vfs)path_openat, which:
  - Traverses directory tree by finding child dir/file w/ (fs)lookup
- **Your job: Implement lookup**

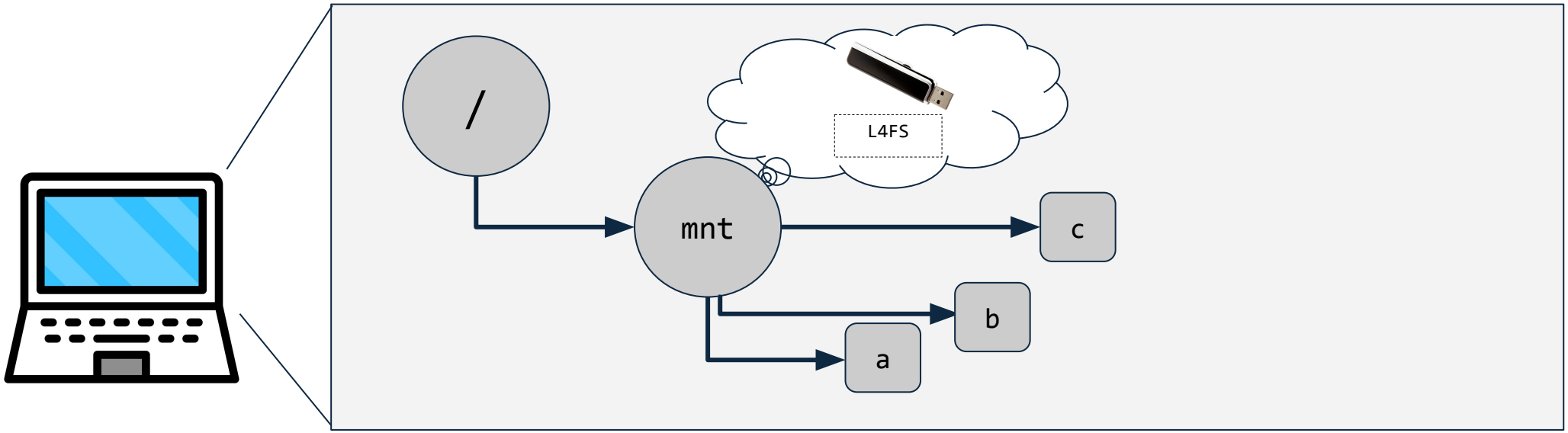
Part 2.2: *creat*



creat /mnt/c

- (syscall)creat -> (vfs)path_openat, which:
  - Traverses directory tree by finding child dir/file w/ (fs)lookup
  - On final step, if lookup() shows file doesn’t exist, call (fs)create
- **Your job: Implement create**

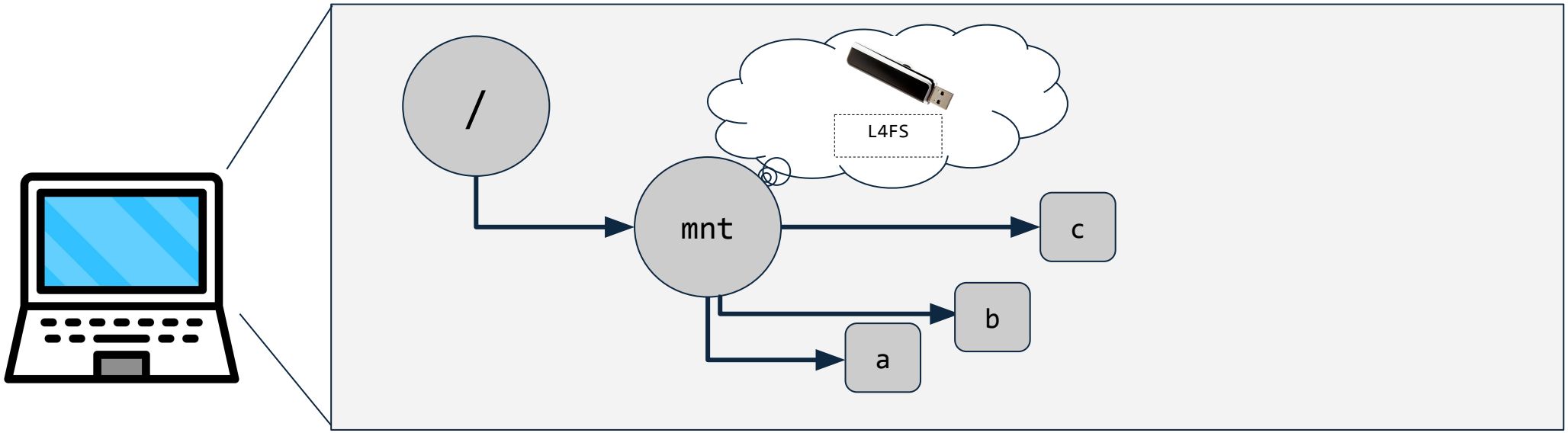
Part 3.1: *read*




```
read(4096KB from a)
```

- `(syscall)read -> (vfs)vfs_read -> (fs)read_iter`
- **Your job: Implement read_iter**

Part 3.2: *write*



write(4096KB to a)

- `(syscall)read -> (vfs)vfs_write -> (fs)write_iter`
- **Your job: Implement write_iter**

Part 4 Overview

- Metadata Journaling
  - Directory structure should be intact
  - Raw data can be lost

- Only for update operations (creat + write)

- In prior example, equiv. to ensuring unplugging USB during write/creat keeps metadata intact

Part 5 Overview

- Criteria for correct concurrent behavior

- Some cases are already accounted for b/c VFS acquires relevant locks
  - You have to figure out which

- Others require explicit locking from you
  - Check VFS locking guide to see which locks to acquire where
  - https://www.kernel.org/doc/html/next/filesystems/locking.html

Sample Questions

- What are L4FS's "on-disk data structures"? How do I read where they are in the code?
- How do I debug while developing?
- Can you give more detail on what Part <x> is asking?
- How do I trace where VFS calls my functions?