2025 OS MP4 - File System

Problem I TA 羅元駿

Problem II TA 陳皓偉

Overview

- Problem I : Access Control & Symbolic Links 40 pts
- Problem II: RAID 1 Simulation 60 pts

Access Control & Symbolic Links

Overall, you have 3 jobs:

- **Symbolic Link**

  Implement `sys_symlink()` in `sysfile.c`, modify `open()` system call.

- **Access control**

  Add permission mode toxv6 file system, implement `sys_chmod()`, modify `open()` system call.

- **Modify ls command**

  Ensure that `ls` shows the mode of each file/directory.

Access Control & Symbolic Links

Symbolic Link

- Command format (fixed, you cannot change it):

```
$ symln old new
```

- System call function prototype (you are free to change it)

```
int symlink(const char *target, const char *path);
```

- We have already added the system call and the command for you.

Access Control & Symbolic Links

Access Control

- Command format (fixed, you cannot change it):

```
$ chmod [-R] (+|-)(r|w|rw|wr) file_name|dir_name
```

- System call format: You decide
- We do not add the `chmod` system call and command for you. You have to add them manually.

Access Control & Symbolic Links

Modify ls command

- Command format:

```
$ ls [file_name|dir_name]
```

- Output format:

```
[file_name]      [type] [inum] [size] [mode]
```

- We do not care [inum] and [size] in test. They can be any number, but it must be present in the output.

Access Control & Symbolic Links

New file type:

- `T_SYMLINK` defined in `kernel/stat.h` with value `4`.
- `ls` a symbolic link should print this value

Access Control & Symbolic Links

New flag for `open`:

- `O_NOACCESS` defined in `kernel/fcntl.h`
- open a file not readable and not writable, but its metadata is accessible by `fstat`
- `open` should not follow the symbolic link when this flag is set; otherwise, `open` should always follow the symbolic link

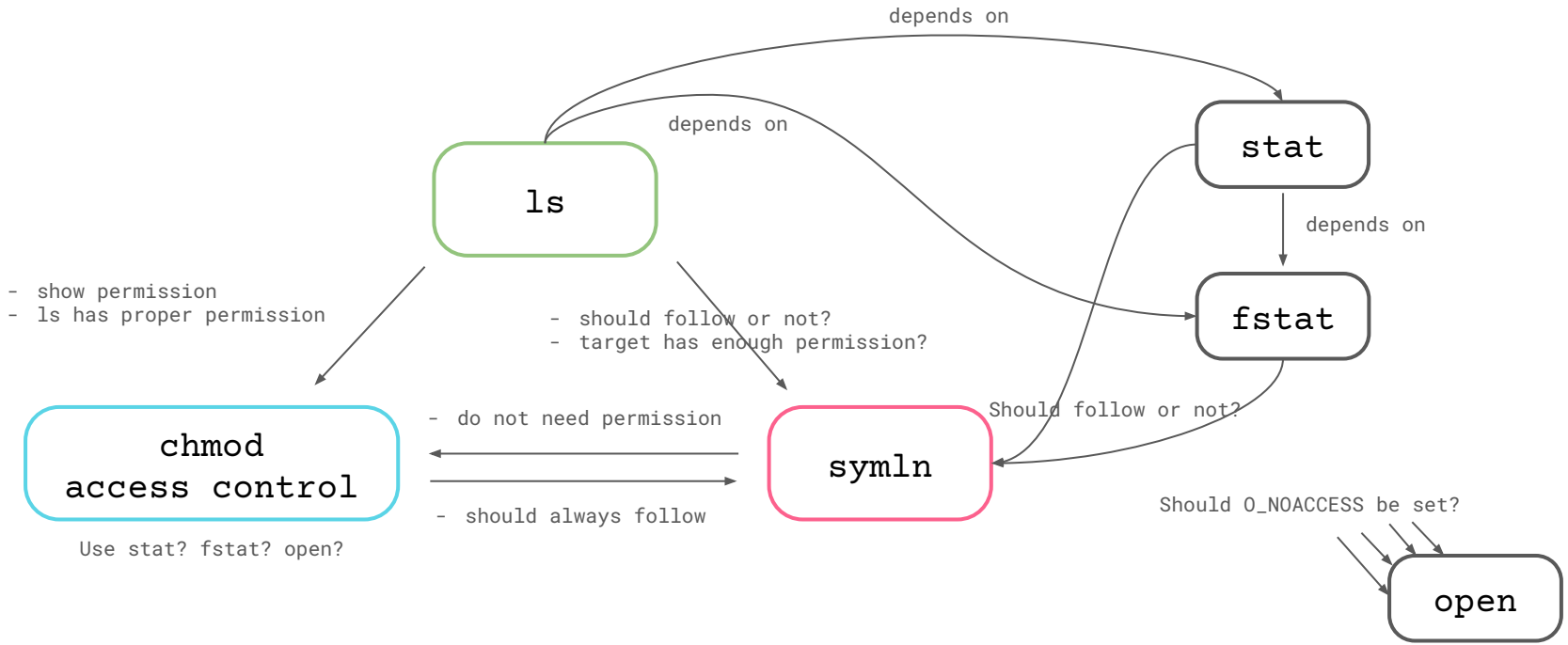
Access Control & Symbolic Links

About test

- **How we test your code:**
  1. Run `gen` command to generate some directory structures.
  2. Run the commands: `symln`, `chmod` and `ls` arbitrary times in arbitrary order and collect the output.
  3. Run `mp4_1` to test your `open` and `fstat` system call.
- **How you test your code:**

```
$ make grade
```

Access Control & Symbolic Links



RAID 1 Simulation

Overall, you have 3 jobs:

- **Mirrored Write**
  Modify bwrite() function in kernel/bio.c to write same data to disk 0 & 1.
- **Write Fallback Logic**
  According to the global variable used to simulate failure, print specific messages to show your logic.
- **Read Fallback Logic**
  Modify bread() function in kernel/bio.c to return the right data if disk or block is failed.

RAID 1 Simulation

Mirrored Write (Modify bwrite)

- The **incoming buffer b** holds the physical block number on Disk 0. Calculate the corresponding block number on Disk 1.
- Set the block number to the buffer before calling I/O function. Reset the block number of the buffer after finishing the I/O.

RAID 1 Simulation

Write Fallback Logic

- Print the specific message provided in the spec while facing different failure.
- The simulated failure is controlled by TA using two parameters:
  - int force_disk_fail_id
  - int force_read_error_pbn.
- **Do not change the name of these parameters!! Or you will get 0 points on this part.**
- **You must follow the format and print the right messages same as those on spec to get the score of this part.**

RAID 1 Simulation

Read Fallback Logic (Modify bread)

- Also check if the block or disk is failed by the two parameters mentioned in the previous page.
- If we want to read the data in the fail block or fail disk, try to get the data from the healthy corresponding block or disk.

RAID 1 Simulation

About test

- **How we test your code:**
  1. Check if the mirrored blocks contain same data.
  2. Check if we fail one block, it can return the right data on another disk.
  3. Check if you print the right format message if we want to write on the fail block or disk.
- **How you test your code:**

```
$ make grade
```

Bonus Report (Optional)

Describe how you help other student with description as short as possible while also as concrete as possible. (e.g. You can screenshot how you help others on NTU COOL)

Submit your report to Bonus Report on Gradescope in one pdf file.

Submission

- **Bonus Report**
  - Submit your report to Gradescope.
- **xv6**
  - Use `diff` to generate a patch file
  - Upload `<student_id>.patch` to NTU COOL
  - Remember to backup your progress, and follow our instruction.