OSLab 5 Report

December 15, 2024

1 Overview

In this lab, we will implement `spawn`, a library call that loads and runs on-disk executables. We will first implement a file system. Then we will implement the `spawn` system call. Finally, we will implement a simple shell that uses the `spawn` system call to run commands.

1.1 Exercise 1

Follow the instructions that give the file system environment the I/O privilege.

1.1.1 Question 1

No need for auxiliary actions to ensure that this privilege is granted to the file system environment. Because the `EFLAGS` register is saved and restored when switching between environments, the file system environment will have the same `EFLAGS` value as the user environment that called it.

1.2 Exercise 2

This two functions are implemented in the file $fs/bc.c$. Follow the instructions and compute the correct address, and call auxiliary functions according to the hints.

1.3 Exercise 3

To allock a new block, we need to find a free block in the block cache. Then we need to immediately flushed the changed bitmap block to disk when we find a free block.

1.4 Exercise 4

To implement the `file_block_walk` function, we first check if the block is directly accessible or not. If it is, we return the block number. Otherwise, we need to find the indirect block and return the block number in the indirect block. To implement the `file_get_block` function, we first get the block number of the file at the given offset using the `file_block_walk` function. Then we need to allocate a new block using the `alloc_block` function.

1.5 Exercise 5&6&7

Follow the instructions.

1.6 Exercise 8

It's the largest challenge in this lab I think. The previous implement of `duppage` is not correct. And it takes me a lot of time to debug it. Moreover, when I implement the previous functions, I set the `file` structure to be `aligned`, which I think make some code wrong. The key point is the wrong right of writing the file. Finally I reimplement the complete lab and it works this time.

1.7 Exercise 9

Simply follow the instructions.

1.8 Exercise 10

The task is simple with a few lines of code. However, I find the shell doesnot work as expected. In fact, all code the at right does not work. For example, the `echo` command and the `num` command work well when executed alone. However, when using `|` to combine them, the shell will hang. Sometimes it reports a pipe competition error. I find a strange assert $n = nn$ in $lib/pipe.c/_pipeisclosed()$, simplt remove it and the shell works well.

1.9 challenge

I implement a block eviction strategy to improve the performance of the file system. I write a function $evict\_policy()$ to go through all blocks except the super block and the bitmap block, and evict all blocks. I use a variable in $bc\_pgfault$ to record the time this function is called. When the number of this variable is greater than a certain value, I call the $evict\_policy()$ function.

1.10 Result

The result is shown in the figure below.

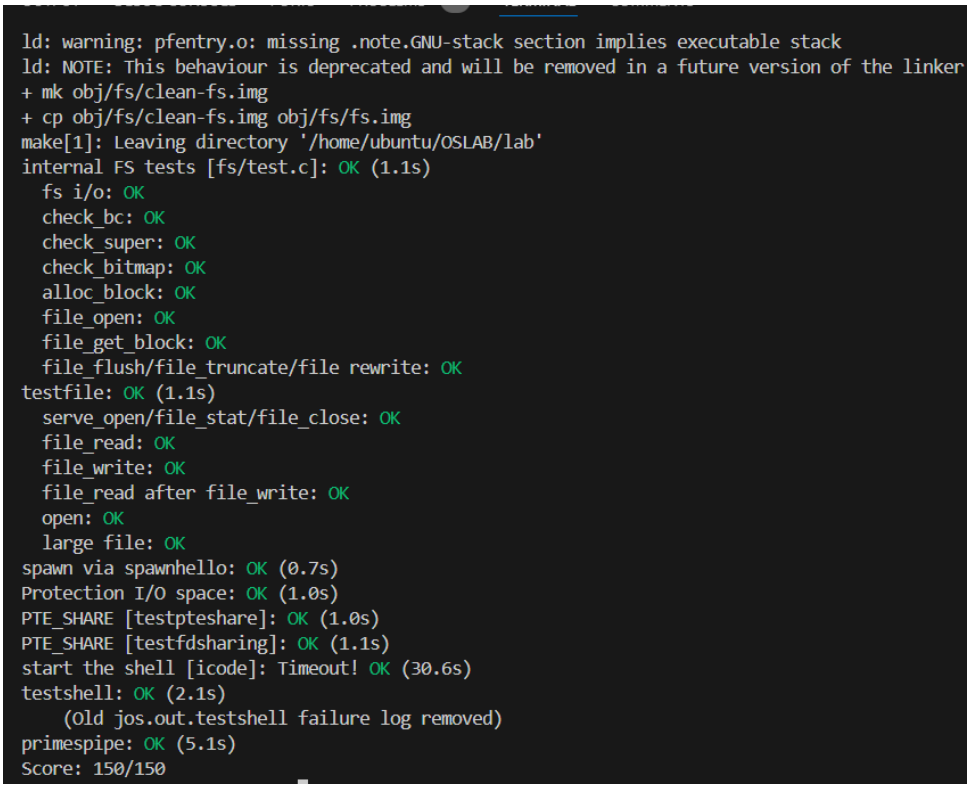

Figure 1: The result of the lab