CS 4375 Fall 2022

Shirley Moore, Instructor

Lab 5 (60 points)

**xv6 File System Edgar Garnica**

In this lab, we will explore how xv6 manages and provides access to files. Since xv6 is a simplified UNIX-like system, this is similar to how files are managed in a production UNIX operating system, although simplified.

Create a lab5 branch of your xv6 repository where you will put the code you write for Task 2c.

Task 1a (10 points)

Look at the ls code in user/ls.c and explain how the code works in terms of the library and system calls it makes. Show the output from the ls command and interpret the different columns.

Text

Description automatically generated

**The code will print like its shows in the screenshot in the order of name, type ,number ,size of the files in the directory(path). The code uses memmove() to get a ponter from the are we are trying to access in this case in print the information.**

Task 1b. (10 points)

Look at the mkdir code in user/mkdir.c and explain how the code works in terms of the system call it makes and the OS helper functions used by that system call. Use mkdir and some other commands (e.g., echo) to create a new directory and a couple of files in it. Run ls again and explain the output.





**The code will execute the argument that is pass after the command is executed and will create the directory with the mkdir() call**

Task 1c. (10 points)

Look at the ln code in user/ln.c and explain how the code works in terms of the library and system calls it makes. Use ln to create one or more links to a file you created in Task 1b. Run ls again and explain the output.

Text

Description automatically generated

**The link() call is used to create a link between an existing file or will create a new file, in the if statement in the code will procced if the call returns zero, which means that the link was created.**

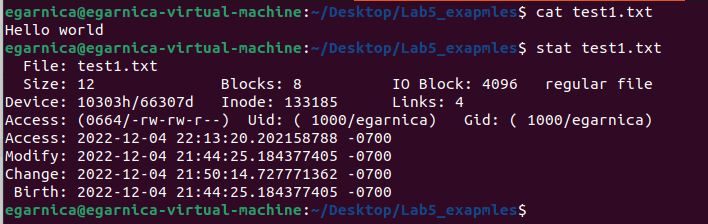
Task 2a. (10 points)

Look at the code in kernel/sysfile.c for the fstat() system call and in kernel/file.c for the filestat() helper function and explain how the codes work.

**The code will use a lock to access the page table, and will use the call of copy out with the address and the size of it. The command will return the information of the desired file.**

Task 2b. (10 points)

On a Linux system, look at the stat man page. Run the stat command on a regular file and on a directory and explain the output.



**The command will return information about a file, no permissions are required on the file itself, but to execute permission is required on all of the directories in path that lead to the file.**

Task 2c. (10 points)

In your lab5 branch, implement an fstat command for xv6 that takes the name of a regular file or a directory as an argument and outputs the information for that file. The information output should include the filename, file type, file size, inode number, and number of links in an easily understandable format.

Sample output: $ fstat fstat name: fstat type: regular file size: 23272 bytes inode number: 31 links: 1

$ mkdir a

$ fstat a

name: a type: directory

size: 32 bytes inode number: 33 links: 1

$ echo "hello" >a/b

$ ln a/b a/c

$ cat a/c

"hello"

$ fstat a/b name: a/b type: regular file size: 8 bytes inode number: 34 links: 2

$ fstat a/c name: a/c type: regular file size: 8 bytes inode number: 34 links: 2

$ cd a

$ ../fstat b

name: b type: regular file

size: 8 bytes inode number: 34 links: 2