Eric Smith
12/02/2022
CS4375

Lab 5 Report

Task 1a:

Task 1 requires us to look at user/ls.c. Running this command gives me a list of all the available user programs which can be run as well as other files or directories. The first column is designated to display the name of the program, the second column is designated to the type of the program, the third column is designated to the inode of the program, and the final column is designated to the size of the program.

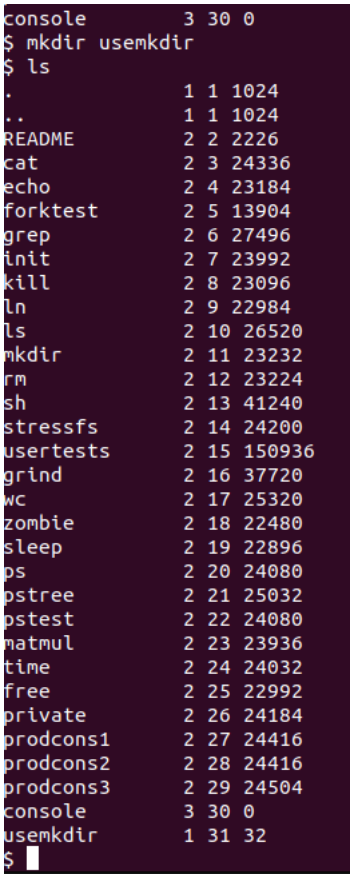
Within the ls code, there are a few initial library calls which are used. One of these library calls is fstat, which displays the file name and status, the open call is used to open a given file, the memmove which is used to move memory from one area to the other, as well as some simple print commands.

The ls user program works by checking if the file containing user programs can be opened and entered as a parameter of fstat. If so, then the program will display information about the file or directory.

```
hart 2 starting
hart 1 starting
init: starting sh
$ ls
.              1 1 1024
..             1 1 1024
README         2 2 2226
cat            2 3 24336
echo           2 4 23184
forktest       2 5 13904
grep           2 6 27496
init           2 7 23992
kill           2 8 23096
ln             2 9 22984
ls             2 10 26520
mkdir          2 11 23232
rm             2 12 23224
sh             2 13 41240
stressfs       2 14 24200
usertests      2 15 150936
grind          2 16 37720
wc             2 17 25320
zombie         2 18 22480
sleep          2 19 22896
ps             2 20 24080
pstree         2 21 25032
pstest         2 22 24080
matmul         2 23 23936
time           2 24 24032
free           2 25 22992
private        2 26 24184
prodcons1      2 27 24416
prodcons2      2 28 24416
prodcons3      2 29 24504
console        3 30 0
$
```

Task 1b:

The mkdir command is a command which makes a directory within the current directory. The code works by checking the arguments and making sure that the parameters are valid. It first makes sure that the directory has a name and then checks if the name of the newly created directory is valid. I am not fully able to verify what makes a valid directory name, but if it works like other operating systems, then certain special characters may be off limits. As with other operating systems, the echo command can either contain other directories or files. The following is an example of the mkdir command.



Going into the usemkdir directory, we can create files or even use mkdir again. Directories create a tree structure in xv6 as well as most other operating systems.

Task 1c:

The ln command works by associating a file name to a given inode number. This gives the appearance that multiple files are duplicated for the user. This is because there are 2 names given for a given file. The first name is the name that the user sees. This can be any name designated by the user. The second name is the inode number. The inode number is the number that the operating system uses to keep track of a file or directory within an operating

system. Inodes have a 1 to one relationship with files, which means that files can only have one inode. Ln allows for multiple files to have the same associated inode.

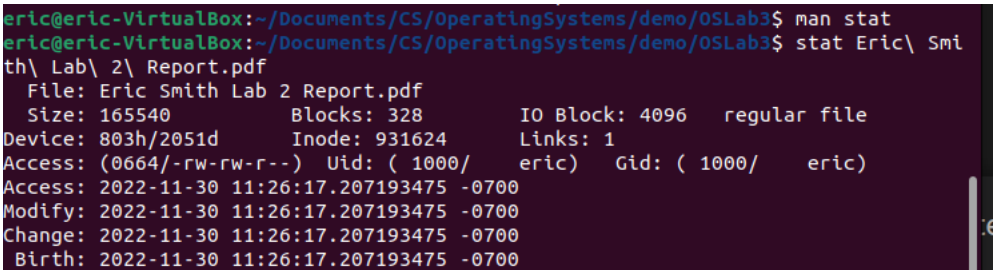
Task 2a:

In this task, we are required to look at sysfile and determine how the systat system call works. When looking at the sysfile.c file in general, it is noted that many checks are made on system call arguments are made in order to make sure that the operating system is protected and safe. The fstat system call starts by making sure that the argument is valid and then calls the filestat helper function.

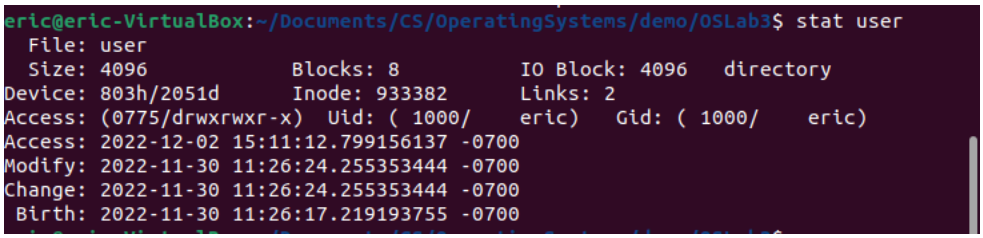
The filestat helper function is relatively simple. It works by checking if the file has an appropriate INODE type. Once it has been asserted that the inode type is valid, an inode lock is acquired for the specified file. A stati system call is made to copy out information about the inode. The lock is then released, and the system call returns successfully. In many ways, this system call is very similar to the getprocs system call we implemented in the first lab.

Task 2b:

In the linux operating system, stat is used to display information about a file or directory. Running the stat file command using a previous lab report, a wide variety of information is displayed about the file. This includes the name, size, IO blocks allocated, creation time, modification time, and permissions of the file.



Running the command on a directory displays very similar information.



Task 2c:

In task 2c, we were required to implement a user program fstat. This program would show information about a program using the filestat system call. Implementing this just required the acquiring the file number with open and passing this to the fstat system call. The system call populates a stat structure which is then printed out.