Lab 5 – Inodes and Protection in Linux

Operating Systems Comp Sci 3SH3, Fall 2024

Prof. Neerja Mhaskar

Lab Format:

1. You are required to show your working solution of this lab to the TA for a grade.
2. The TA will check your solution and will quiz you on your work. After which they will enter your mark and feedback on Avenue.
3. If you do not show your work to your Lab TA, you will get a zero (unless you provide an MSAF).
4. It is your responsibility to connect with your Lab TA to get a grade and ensure that your grade has indeed been posted on Avenue.

During your lab hours, the TAs will also be available to answer any questions you may have on your assignments.

Outline

This practice lab focuses on an important data structure used to represent files in a system called **file control blocks**. In UNIX and LINUX systems the file control blocks are called **inodes**. An inode stores the attributes and disk block location(s) of the respective file/directory on the system.

In these practice labs you will learn the following:

1. Obtain file attributes using **`ls -li`** and **`stat`** commands.
2. Learn to use the **`stat`** data structure to output file attributes in C.
3. Learn about **`script`** and **`exit`** commands

Part I - Obtaining file attributes using `ls -li or stat` command.

1. Create a simple text file named file1.txt.
2. Add the following line "This is file 1" to the file and save it.
3. Obtain the inode number and other few file attributes of file1.txt with the following command:

```
ls -li file1.txt
```

4. On my machine this produces output similar to the following:

```
400740 -rw-r--r-- 1 oscreader oscreader 15 Mar 24 18:52 file1.txt,
```

where the number in bold is the inode number of the file. Note: The inode number of file1.txt is likely to be different on your system.

5. You can obtain detailed file attributes and the inode number with the **`stat`** command.

```
stat file1.txt
```

6. This produces output similar to the following:

```
File: ‘file1.txt’
Size: 15 Blocks: 16 IO Block: 4096 regular file
Device: 801h/2049d Inode: 400740 Links: 1
Access: (0644/-rw-r--r--) Uid: ( 1000/oscreader)
Gid: ( 1000/oscreader)
Access: 2017-03-24 18:52:30.776000000 -0600
Modify: 2017-03-24 18:52:30.628000000 -0600
Change: 2017-03-24 18:52:30.760000000 -0600
Birth: -
```

PART II - Using the **`stat`** data structure to output file attributes in C

You are to write a C program (named lab5.c) that takes the text file **`file1.txt`** you created before, as a command line argument and outputs the following file attributes stored in the input file’s inode:

1. Inode number
2. File size
3. Blocks
4. User ID
5. File permissions (in the same format as seen in the output of stat command)
6. Time of last access
7. Time of last data modification
8. Last status change time

In particular your C program needs to do the following:
1. Use the **`stat`** data structure to output file attributes. More information about stat structure can be found at:

http://pubs.opengroup.org/onlinepubs/7908799/xsh/sysstat.h.html

2. To be able to use the **`stat`** structure add the following header files:

```
#include <stdlib.h>
#include <stdio.h>
```

```
#include <errno.h>
#include <string.h>
#include <sys/types.h>
#include <sys/stat.h> /*Contains the stat structure definition*/
```

3. Declare a variable of type **`stat`**: E.g.:

```
struct stat fileAttributes;
```

4. Use the following members of **`stat`** to print the required file attributes.

a. Inode: `fileAttributes.st_ino`
b. Size (in bytes): `fileAttributes.st_size`
c. Blocks: `fileAttributes.st_blocks`
d. File Permissions: `fileAttributes.st_mode`
e. Uid: `fileAttributes.st_uid`
f. Time of last access: `fileAttributes.st_atime`
g. Time of last data modification: `fileAttributes.st_mtime`
h. Last Status Change time: `fileAttributes.st_ctime`

To print the file permissions in the format as seen in the output of **`stat`** command use the file mode bits **`S_IFDIR, S_IRUSR`** and so on: Sample code:

```
fileAttributes.st_mode & S_IFDIR
fileAttributes.st_mode & S_IRUSR // And so on...

/*
NOTE
You can also use the following to determine if an object is a file
or folder: S_ISDIR(fileAttributes.st_mode)
 */
```

5. Compile your program without errors.

Part-III - Using `script` and `exit` command to log activity on the terminal

*About **`Script`** and **`exit`** command*

`script` command is used to take a copy of everything which is output to the terminal and place it in a log file. The script command should be followed by the name of the log file (e.g. **`script`** output.txt).
`exit` command stops logging to a file initiated by the **`script`** command and closes the file.

After you have your C program working, you are to log the output of the following commands using the **`script`** command. (See sample output.txt file.)

1. **`gcc -o lab5 lab5.c`**
2. **`./lab5 file1.txt`**
3. **`Use chmod`** command to change the permissions on the file1.txt file. You are to change the permissions on the file ‘file1.txt’ as follows:
    a. Owner/user account has Read and Write permissions.
    b. Group account has Read and Write permissions.
    c. Others account has only Read permission.
4. **`./lab5 file1.txt`**
5. **`stat file1.txt`**

Note: see lecture notes on file system to use `chmod` command.

<u>Sample output recorded by script command:</u>

See output.txt file posted on Avenue -> content -> Labs -> Lab5