COSC 301: Operating Systems

Lab 8: Disks and files

1. Assume that you run the following Linux command to query your SCSI disk for its "geometry":

   ```
   $ sudo sfdisk  -g /dev/sda
   dev/sda: 1044 cylinders, 255 heads, 63 sectors/track
   ```

   Further, assume that the average latency (seek time + rotational delay) for this disk is 10 milliseconds (0.01 seconds), and that the disk-to-computer interface can support up to 300 MB/s. (Recall that 1 MB is 2^20 bytes.)

   a. Assuming the sector size is 512 bytes, how large is the disk, in *bytes*?
   b. What is the total latency in milliseconds for reading 1 sector (512 bytes) (to 3 decimal places, with rounding)?
   c. What is the effective bandwidth (in MB/s) achieved for reading 1 sector?
   d. What is the total latency in milliseconds for reading 10240 consecutive sectors (5 MB) (i.e., a *sequential* read)? (Again, to 3 decimal places, with rounding.)
   e. What is the effective bandwidth for performing the 1 MB sequential read?

2. Roll your own `touch`. Write a short C program that implements the behavior of `touch` when there is no pre-existing file. Simply put, you should create an empty file of the given name. You should be able run your `touch` clone like:

   ```
   $ ./mytouch junk.txt
   ```

   The following system calls will be useful:

   - `creat`
   - `close`

   You can assume that the initial permissions for the file should be `0700`. This program shouldn't be any more than about 15-20 lines, total.

3. Roll your own (very basic) `ls`. Write a short program that takes a directory name as a parameter and implements the basic `ls` functionality by printing the contents of the directory. At minimum, your program should print out each entry, whether the entry is a file or directory, and the size (if a file).

   The following system calls will be useful:

   - `opendir`
   - `closedir`
   - `readdir`
   - `stat`

   This program shouldn't be any more than 40 lines or so.

4. `strace` is a neat program that can trace system call usage in other programs.

   Compile your `touch` and `ls` programs in the following way:

```
gcc -static -o mytouch mytouch.c -Wall -g
gcc -static -o myls myls.c -Wall -g
```

The `-static` flag tells `gcc` to compile all libraries "statically" (and use no DLLs, or shared libraries).

Now, type `strace ./mytouch junk.txt`. You should see the system calls that you wrote be invoked, along with some other system calls before and after. Do the same for your `ls` program --- invoke it using `strace`.

Briefly discuss what you see in the `strace` output for each program. You don't ned to describe *every* system call; just give a brief description of what's going on in the OS in order to do what your programs are asking the OS to do.

Before you leave lab

1. Demo your `touch` and `ls` clones for me (working demo and code)
2. Post your answers to problems 1 and 4 on Moodle.