New York University
Tandon School of Engineering
Department of Electrical & Computer Engineering

Introduction to Operating Systems (CS-GY6233)
Fall 2022

Assignment 4
(10 points)

In this assignment, we shall build on what we did in assignment 3 and further extend it. You shall write a dynamically loadable module that registers itself as a character device and allows us to get the current time (which we already implemented in the previous assignment) when that device is read from a user-mode application.

Indeed the "libc" library contains API functions that allows us to obtain the current time but we are going to implement our own version instead, that's the goal of this assignment.

When reading from the module, you may either develop a user-mode program that opens and reads from your device (just as if it's a file) or you may use the following command:

>> cat /dev/lab4

it shall return:

"Hello world, the current time is hh:mm:ss".

Your module should:

a) Use the `misc_register()` functionality to register itself as a character device. This requires that you create a structure (`struct file_operations`) containing the function pointers (open, release, read, etc.) and pass it to `misc_register()`.
b) You may refer to http://www.linuxjournal.com/article/2920 for detailed info on how to create and register a misc device.
c) Define the open(), release() and read() functions. Make sure you populate their function pointers in `struct file_operations`.
d) You may test your code by either writing your own user-mode program that reads from your device (useful for debugging) or by using:

```
>> cat /dev/lab4 // that's if you named your device lab4 during misc_reg()
```

Some Linux Notes:

- Devices are organized as character, block or network devices.

- Devices have major and minor numbers. The major number specifies the device type whereas the minor number specifies an instance of that type. As such, it is common that a single device driver takes care of an entire major number.
- Most major numbers are already assigned to specific device drivers so you are not supposed to use them.
- Linux uses the concept of virtual file systems, one of those is the devfs.
    - Generally, you register your device with the devfs using a major and a minor number.
    - You can then create a node for it in the /dev directory (using the mknod command), e.g. /dev/lab4.
    - User-mode applications may then treat it as a file and file operations such as fopen(), fprint, fread, etc. are routed to functions that your module handles.
- Instead of using this complex registration and node creation process, we shall use the `misc_register()`, which handles both steps automatically. It registers us under device major number 10 (amongst with many other devices besides us using the same major number, where each has a different minor number).
    - You need to provide a minor number to `misc_register()`, the device's name ("lab4") and a pointer to the file operations.
    - In order to avoid collisions, use the `MISC_DYNAMIC_MINOR` macro as the minor number when you register your device.
- For information regarding misc_register(), see: http://www.linuxjournal.com/article/2920.
- You may find further information at kernel.org or lwn.net
- For reasons that shall be explained when we study virtual memory (towards lectures 8-9), you cannot directly use the pointers provided by the read() function in `struct_file_operstions`. Instead you need to use:

```
unsigned long copy_to_user(void __user *to, const void *from, unsigned long n);
```

Submission file structure:

Please submit a **single .zip file** named **[Your Netid]_lab#.zip**. It shall have the following structure (replace # with the actual assignment number):

└── [Your Netid] hw# (Single folder includes all your submissions)

- ├── lab#_1.c (Source code for problem 1)
- ├── lab#_2a.c (Source code for problem 2a, and so on)
- ├── lab#_1.h (Source code header file, if any)
- ├── Makefile (makefile used to build your program, if any)
- ├── lab#.pdf (images + Report/answers to short-answer questions)

What to hand in (using Brightspace):

- Source files (.c or .h) with appropriate comments.
- Your Makefile if any.
- A .pdf file named **"lab#.pdf"** (# is replaced by the assignment number), containing:

- Screen shot(**s**) of your terminal window showing the current directory, the command used to compile your program, the command used to run your program and the output of your program.

RULES:

- You shall **use kernel version 4.x.x or above**. You shall not use kernel version 3.x.x.
- You may consult with other students about GENERAL concepts or methods but copying code (or code fragments) or algorithms is NOT ALLOWED and is considered cheating (whether copied form other students, the internet or any other source).
- If you are having trouble, please ask your teaching assistant for help.
- You must submit your assignment prior to the deadline.