Lab 1- Process Information- Group 12

Problem Definition

The solution created in this lab experiment focuses on the development of a loadable kernel module for the Linux kernel to provide information on the currently running process in the process control block. This information consists of the Process ID (PID), the Parent Process ID (PPID), the state of the current task, and several group/user identifiers (GID/UID). The module creates a file in the /proc directory named “lab1” which contains all the relevant information of the current process which can be displayed to the user as seen in the output.

The code starts out with various Linux kernel headers required for module development. It then checks the Linux kernel version to determine which structure is better for handling /proc operations. If the kernel version is earlier than 5.6, it uses struct file_operations, or else it uses struct proc_ops.

The module contains several segments of code, including the lab1_init function which initializes the kernel module and allows the user to create an entry into the /proc/lab1 directory using the proc_create function, where it then prints a message to the kernel log to indicate that the module has been loaded. If the creation of the /proc/lab1 entry fails, it returns an error code.

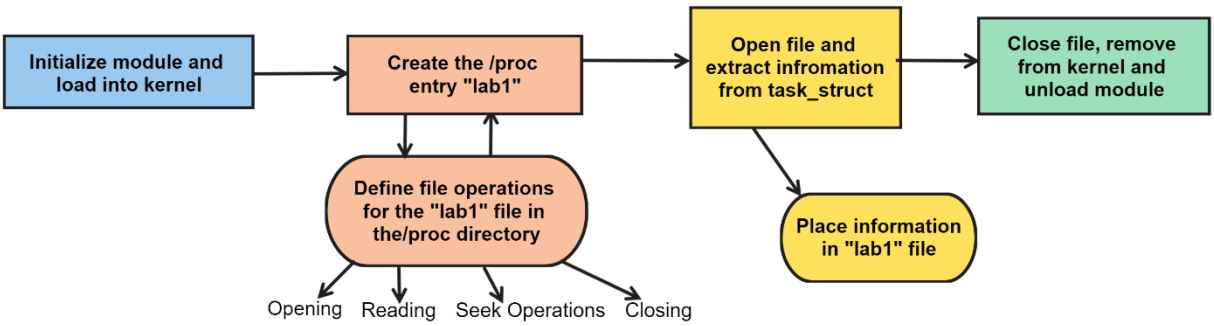
Code was written to handle file operations using two structure definitions (struct proc_ops and struct file_operations) which exist to handle multiple versions of the Linux kernel. These functions allow for the opening, reading, handling of seek operations, and closing of files.

The lab1_show function extracts information from the current process, by accessing a pointer to the current task_struct. This then points to the various information within the task_struct, allowing for desired information to be written to the output /proc/lab1 file using the seq_file interface. This section also includes conditionals that assesses and outputs the current state of the task, identifying whether it is running, waiting or stopped.

The lab1_open function is called when a user space process tries to open the /proc/lab1 file. It sets up the reading of the file by calling single_open, which will use the lab1_show function to produce the file's content.

The lab1_exit function removes the entry from the proc directory and indicates to the user that the module has been removed from the kernel’s memory and its execution has been stopped. The following flow diagram describes the module's processes.

The final three lines of code specify module information. It shows that the module is licensed under the GPL license, and registers the initialization and exit functions for the module.


Initialize module and load into kernel
Create the /proc entry "lab1"
Open file and extract infromation from task_struct
Close file, remove from kernel and unload module
Define file operations for the "lab1" file in the/proc directory
Place information in "lab1" file
Opening
Reading
Seek Operations
Closing