Lab 1 Report

Name: Insiyah Ujjainwala
Email: ujjainwi@mcmaster.ca
Student ID: 400357483
Date: 29/09/2023

Note:
I completed this lab and showed my work to my TA Nishant during my lab on Wednesday at 8:30am. After which my terminal became unresponsive and I wasn't able to access any of my work. In an attempt to resolve this, I restarted my laptop but somehow that made things worse by completely uninstalling the virtual machine that had my files stored. I was able to rewrite the .c files with the code but I kept running into different issues while trying to compile my code and print the output. This is why I humbly request you to overlook the missing outputs in the lab report. I have also consulted Nishant for the same issue and he advised I should submit the code without output and it should be okay. As for the errors, he will be helping me resolve them during our lab on Monday to ensure this situation does not arise in the future.

jiffies.ko

- The purpose of this module is to print the current module of jiffies.
- We first run the ‘make’ command to create the jiffies.ko file.
- After which we run ‘sudo insmod jiffies.ko’ to load the kernel module.
- Finally, we run the cat /proc/jiffies command to create a new entry in the /proc file system that in turn prints the value of jiffies to our terminal.

Code:

```
/**
* jiffies.c
*
* Kernel module that prints the current value of jiffies
* Prints this value when cat /proc/jiffies command is called.
*
* ujjainwi@mcmaster.ca
*/

// For kernel programming
#include <linux/init.h>
#include <linux/module.h>
#include <linux/kernel.h>
#include <asm/uaccess.h>

// For proc read
#include <linux/proc_fs.h>

// For jiffies
#include <linux/jiffies.h>

// Defines a constant buffer size for the string to be printed when cat
/proc/seconds is called.
#define BUFFER_SIZE 128

// Defines the name of the proc
#define PROC_NAME "jiffies"

static ssize_t proc_read(struct file *file, char __user *usr_buf, size_t count,
loff_t *pos);

static struct proc_ops proc_ops= {
   //.owner = THIS_MODULE,
   .proc_read = proc_read,
};
```

```c
// This function is called when the module is loaded.
static int proc_init(void)
{
   // creates the /proc/jiffies entry
   // the following function call is a wrapper for
   // proc_create_data() passing NULL as the last argument
   proc_create(PROC_NAME, 0666, NULL, &proc_ops);
   printk(KERN_INFO "/proc/%s created\n", PROC_NAME);

   return 0;
}

// This function is called when the module is removed.
static void proc_exit(void) {

   // removes the /proc/jiffies entry
   remove_proc_entry(PROC_NAME, NULL);
   printk( KERN_INFO "/proc/%s removed\n", PROC_NAME);
}

// This function is called each time the /proc/jiffies is read.

static ssize_t proc_read(struct file *file, char __user *usr_buf, size_t count,
loff_t *pos)
{
   // Use rv to store the number of characters printed
   int rv = 0;
   char buffer[BUFFER_SIZE];
   static int completed = 0;

   // Use complete as boolean to prevent repeat reads
   if (completed) {
       completed = 0;

       // Return 0 to prevent repeat reads if completed
       return 0;
   }

   // Set to 1 after proc is read
   completed = 1;

   // Store the number of characters printed in rv
   // sprintf returns the number of characters in the buffer
   rv = sprintf(buffer, "Current value of jiffies: %lu\n", jiffies);
```

```
    // copies the contents of buffer to usr_buf
    copy_to_user(usr_buf, buffer, rv);

    // Return the printed string length
    return rv;
}


// Set the module entry and exit points. */
module_init( proc_init );
module_exit( proc_exit );

MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("Kernel module to print current value of jiffies");
MODULE_AUTHOR("Insiyah");
```

seconds.ko

- The purpose of this module is to print the seconds elapsed since the kernel module was loaded.
- We first run the ‘make’ command to create the seconds.ko file.
- After which we run ‘sudo insmod seconds.ko’ to load the kernel module.
- Finally, we run the cat /proc/seconds command to create a new entry in the /proc file system that in turn prints the value of jiffies to our terminal.

Code:

```
/**
* seconds.c
*
* Kernel module that calculates and prints the seconds that have
* elapsed since the kernel module was loaded.
* Prints this value when cat /proc/seconds command is called.
*
* ujjainwi@mcmaster.ca
*/

// For kernel programming
#include <linux/init.h>
#include <linux/module.h>
#include <linux/kernel.h>
```

```c
#include <asm/uaccess.h>

// For proc read
#include <linux/proc_fs.h>

// For jiffies
#include <linux/jiffies.h>

// Defines a constant buffer size for the string to be printed when cat
/proc/seconds is called.
#define BUFFER_SIZE 128

// Defines the name of the proc
#define PROC_NAME "seconds"

static ssize_t proc_read(struct file *file, char __user *usr_buf, size_t count,
loff_t *pos);

static struct proc_ops proc_ops= {
   //.owner = THIS_MODULE,
   .proc_read = proc_read,
};

// This function is called when the module is loaded.
static int proc_init(void)
{
   // Stores the initial value of jiffies
   long int jiffies = jif;

   // creates the /proc/seconds entry
   // the following function call is a wrapper for
   // proc_create_data() passing NULL as the last argument
   proc_create(PROC_NAME, 0666, NULL, &proc_ops);
   printk(KERN_INFO "/proc/%s created\n", PROC_NAME);

   return 0;
}

// This function is called when the module is removed.
static void proc_exit(void) {

   // removes the /proc/seconds entry
   remove_proc_entry(PROC_NAME, NULL);
   printk( KERN_INFO "/proc/%s removed\n", PROC_NAME);
}
```

```c
// This function is called each time the /proc/seconds is read.

static ssize_t proc_read(struct file *file, char __user *usr_buf, size_t count,
loff_t *pos)
{
   // Use rv to store the number of characters printed
   int rv = 0;
   char buffer[BUFFER_SIZE];
   static int completed = 0;

   // Use complete as boolean to prevent repeat reads
   if (completed) {
       completed = 0;

       // Return 0 to prevent repeat reads if completed
       return 0;
   }

   // Set to 1 after proc is read
   completed = 1;

   // Store the number of characters printed in rv
   // sprintf returns the number of characters in the buffer
   rv = sprintf(buffer, "Seconds elapsed: %lu\n", ((jiffies - jif)/HZ));

   // copies the contents of buffer to usr_buf
   copy_to_user(usr_buf, buffer, rv);

   // Return the printed string length
   return rv;
}


// Set the module entry and exit points. */
module_init( proc_init );
module_exit( proc_exit );

MODULE_LICENSE("GPL");
MODULE_DESCRIPTION("Kernel module to find seconds elapsed");
MODULE_AUTHOR("Insiyah");
```