作業系統 #HW1 Programm... HackMD (https://hackmd.io?utm_source=view-page&utm_medium=logo-nav)

作業系統 #HW1 Programming Projects

Team Member

- 資工三 110590011 劉承軒 - 程式整理、填寫文件
- 資工三 110590018 劉承翰 - 環境架設、程式撰寫
- 資工三 110590056 林星主 - 資料查詢、填寫文件

Chap.2 Linux Kernel Modules

Description

1. Design a kernel module that creates a /proc file named /proc/jiffies that reports the current value of jiffies when the /proc/jiffies file is read, such as with the command: `cat /proc/jiffies`
    - Be sure to remove /proc/jiffies when the module is removed.
2. Design a kernel module that creates a proc file named /proc/seconds that reports the number of elapsed seconds since the kernel module was loaded.
    - This will involve using the value of jiffies as well as the HZ rate.
    - When a user enters the command `cat /proc/seconds` your kernel module will report the number of seconds that have elapsed since the kernel module was first loaded.
    - Be sure to remove /proc/seconds when the module is removed.

Environment

Ubuntu: 20.04.6 LTS(Kernel: Linux 5.15.0-101-generic)
Compiler: GCC 9.4.0

Details

jiffies.c

```c
#include <linux/init.h>
#include <linux/module.h>
#include <linux/proc_fs.h>
#include <linux/jiffies.h>
#include <linux/uaccess.h>  // Needed for copy_to_user()

#define PROC_NAME "jiffies"

static struct proc_dir_entry *proc_file;

static ssize_t proc_read_jiffies(struct file *file, char __user *buffer,
    char temp[20];
    int len;

    len = snprintf(temp, sizeof(temp), "%lu\n", jiffies);
    if (*offset >= len)
        return 0;

    if (copy_to_user(buffer, temp, len))
        return -EFAULT;

    *offset += len;
    return len;
}

static const struct proc_ops proc_fops = {
    .proc_read = proc_read_jiffies,
};

static int __init proc_init(void) {
    proc_file = proc_create(PROC_NAME, 0444, NULL, &proc_fops);
    printk(KERN_INFO "/proc/%s created\n", PROC_NAME);
    if (!proc_file) {
        return -ENOMEM;
    }
    return 0;
}

static void __exit proc_exit(void) {
    remove_proc_entry(PROC_NAME, NULL);
    printk(KERN_INFO "/proc/%s removed\n", PROC_NAME);

}

module_init(proc_init);
module_exit(proc_exit);

MODULE_LICENSE("GPL");
MODULE_AUTHOR("orange");
```

second.c

```c
#include <linux/init.h>
#include <linux/module.h>
#include <linux/proc_fs.h>
#include <linux/jiffies.h>
#include <linux/time.h>
#include <linux/uaccess.h>  // Needed for copy_to_user()

#define PROC_NAME "seconds"

static struct proc_dir_entry *proc_file;
static unsigned long start_jiffies;

static ssize_t proc_read_seconds(struct file *file, char __user *buffer,
    unsigned long elapsed_seconds = (jiffies - start_jiffies) / HZ;
    char temp[20];
    int len;

    len = snprintf(temp, sizeof(temp), "%lu\n", elapsed_seconds);
    if (*offset >= len)
        return 0;

    if (copy_to_user(buffer, temp, len))
        return -EFAULT;

    *offset += len;
    return len;
}

static const struct proc_ops proc_fops = {
    .proc_read = proc_read_seconds,
};

static int __init proc_init(void) {
    start_jiffies = jiffies;
    proc_file = proc_create(PROC_NAME, 0444, NULL, &proc_fops);
    printk(KERN_INFO "/proc/%s created\n", PROC_NAME);
    if (!proc_file) {
        return -ENOMEM;
    }
    return 0;
}

static void __exit proc_exit(void) {
    remove_proc_entry(PROC_NAME, NULL);
    printk(KERN_INFO "/proc/%s removed\n", PROC_NAME);

}

module_init(proc_init);
module_exit(proc_exit);

MODULE_LICENSE("GPL");
MODULE_AUTHOR("orange");
```

MakeFile

```
obj-m += jiffies.o seconds.o
all:
	make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules
clean:
	make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean
```

Execution Method

1. 打開 Terminal
2. 使用 `cd` 指令到 jiffies.c 和 second.c 的目錄中
3. 使用 `make` 指令執行 makefile，輸出 jiffies.ko 和 second.ko 檔案
4. 裝載模塊 `sudo insmod jiffies.ko` 和 `sudo insmod second.ko`
5. 讀取 /proc `cat /proc/jiffies.ko` 和 `cat /proc/second.ko`
6. 卸載模塊 `sudo rmmod jiffies.ko` 和 `sudo rmmod second.ko`

Result

```
orange@ubuntu:~/Desktop/OS/LinuxKernelModules$ make
make -C /lib/modules/5.15.0-101-generic/build M=/home/orange/Desktop/OS/LinuxKernelModules modules
make[1]: Entering directory '/usr/src/linux-headers-5.15.0-101-generic'
  CC [M]  /home/orange/Desktop/OS/LinuxKernelModules/jiffies.o
  CC [M]  /home/orange/Desktop/OS/LinuxKernelModules/seconds.o
  MODPOST /home/orange/Desktop/OS/LinuxKernelModules/Module.symvers
  CC [M]  /home/orange/Desktop/OS/LinuxKernelModules/jiffies.mod.o
  LD [M]  /home/orange/Desktop/OS/LinuxKernelModules/jiffies.ko
  BTF [M] /home/orange/Desktop/OS/LinuxKernelModules/jiffies.ko
Skipping BTF generation for /home/orange/Desktop/OS/LinuxKernelModules/jiffies.ko due to unavailability of vmlinux
  CC [M]  /home/orange/Desktop/OS/LinuxKernelModules/seconds.mod.o
  LD [M]  /home/orange/Desktop/OS/LinuxKernelModules/seconds.ko
  BTF [M] /home/orange/Desktop/OS/LinuxKernelModules/seconds.ko
Skipping BTF generation for /home/orange/Desktop/OS/LinuxKernelModules/seconds.ko due to unavailability of vmlinux
make[1]: Leaving directory '/usr/src/linux-headers-5.15.0-101-generic'
orange@ubuntu:~/Desktop/OS/LinuxKernelModules$ sudo insmod jiffies.ko
orange@ubuntu:~/Desktop/OS/LinuxKernelModules$ sudo insmod seconds.ko
orange@ubuntu:~/Desktop/OS/LinuxKernelModules$ cat /proc/jiffies
4297007775
orange@ubuntu:~/Desktop/OS/LinuxKernelModules$ cat /proc/seconds
13
orange@ubuntu:~/Desktop/OS/LinuxKernelModules$ dmesg | tail
[ 7392.830369] perf: interrupt took too long (78472 > 77121), lowering kernel.perf_event_max_sample_rate to 2500
[ 7417.474501] systemd-rc-local-generator[72847]: /etc/rc.local is not marked executable, skipping.
[ 7418.203737] systemd-rc-local-generator[72908]: /etc/rc.local is not marked executable, skipping.
[ 7419.234998] systemd-rc-local-generator[72941]: /etc/rc.local is not marked executable, skipping.
[ 7419.763402] systemd-rc-local-generator[72984]: /etc/rc.local is not marked executable, skipping.
[ 7420.181695] systemd-rc-local-generator[73013]: /etc/rc.local is not marked executable, skipping.
[ 7420.668708] systemd-rc-local-generator[73051]: /etc/rc.local is not marked executable, skipping.
[ 7601.593005] systemd-rc-local-generator[84099]: /etc/rc.local is not marked executable, skipping.
[ 8446.436044] /proc/jiffies created
[ 8455.540358] /proc/seconds created
orange@ubuntu:~/Desktop/OS/LinuxKernelModules$ sudo rmmod jiffies
orange@ubuntu:~/Desktop/OS/LinuxKernelModules$ sudo rmmod seconds
orange@ubuntu:~/Desktop/OS/LinuxKernelModules$ dmesg | tail
[ 7418.203737] systemd-rc-local-generator[72908]: /etc/rc.local is not marked executable, skipping.
[ 7419.234998] systemd-rc-local-generator[72941]: /etc/rc.local is not marked executable, skipping.
[ 7419.763402] systemd-rc-local-generator[72984]: /etc/rc.local is not marked executable, skipping.
[ 7420.181695] systemd-rc-local-generator[73013]: /etc/rc.local is not marked executable, skipping.
[ 7420.668708] systemd-rc-local-generator[73051]: /etc/rc.local is not marked executable, skipping.
[ 7601.593005] systemd-rc-local-generator[84099]: /etc/rc.local is not marked executable, skipping.
[ 8446.436044] /proc/jiffies created
[ 8455.540358] /proc/seconds created
[ 8493.861360] /proc/jiffies removed
[ 8500.578730] /proc/seconds removed
```

Chap.3 Project 2: Linux Kernel Module for Task Information

Description

- In this project, you will write a kernel module that uses /proc file system for displaying a task's information in a Linux system.
  - Be sure to review the programming project in Chap.2 before you begin this project
  - It can be completed using the Linux virtual machine provided with the textbook
- Part I – Writing to the /proc file system
- Part II – Reading from the /proc file system

Environment

Ubuntu: 20.04.6 LTS(Kernel: Linux 5.15.0-101-generic)
Compiler: GCC 9.4.0

Details

proc_read()

```
static ssize_t proc_read(struct file *file, char __user *usr_buf, size_t
    char buffer[BUFFER_SIZE];
    int len;
    static int completed = 0;
    struct task_struct *task;
    char task_state ;


    if (completed) {
        completed = 0;
        return 0;
    }

    completed = 1;

    task = pid_task(find_get_pid(current_pid), PIDTYPE_PID);
    if (!task) {
        printk(KERN_INFO "Task with PID %d not found.\n", current_pid);
        return -ESRCH;
    }
    task_state = task_state_to_char(task);
    len = snprintf(buffer, BUFFER_SIZE, "Command: %s\nPID: %d\nState: %c\
                  task->comm, current_pid, task_state);

    if (copy_to_user(usr_buf, buffer, len)) {
        return -EFAULT;
    }

    return len;
}
```

proc_write()

```
static ssize_t proc_write(struct file *file, const char __user *usr_buf,
    char k_buf[BUFFER_SIZE];

    if (count > BUFFER_SIZE) count = BUFFER_SIZE;
    if (copy_from_user(k_buf, usr_buf, count)) return -EFAULT;

    sscanf(k_buf, "%d", &current_pid);

    return count;
}
```

MakeFile

```
obj-m += taskinfo.o
all:
	make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules
clean:
	make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean
```

Execution Method

1. 打開 Terminal
2. 使用 `cd` 指令到 taskinfo.c 的目錄中
3. 使用 `make` 指令執行 makefile，輸出 taskinfo.ko 檔案
4. 裝載模塊 `sudo insmod taskinfo.ko`
5. 寫入 /proc `echo 1 > /proc/taskinfo.ko`
6. 讀取 /proc `cat /proc/taskinfo.ko`
7. 卸載模塊 `sudo rmmod taskinfo.ko`

Result

```
orange@ubuntu:~/Desktop/OS$ sudo insmod taskinfo.ko
orange@ubuntu:~/Desktop/OS$ echo 1 > /proc/taskinfo
orange@ubuntu:~/Desktop/OS$ dmesg | tail
[ 2679.580423] /proc/taskinfo created
[ 2711.581575] Task with PID 0 not found.
[ 2754.301183] perf: interrupt took too long (39257 > 39202), lowering kernel.perf_event_max_sample_rate to 5000
[ 3022.353204] Task with PID 1234 not found.
[ 3029.567966] Task with PID 1234 not found.
[ 3032.269808] Task with PID 1234 not found.
[ 3269.468475] perf: interrupt took too long (49221 > 49071), lowering kernel.perf_event_max_sample_rate to 4000
[ 3330.355171] /proc/taskinfo removed
[ 3816.317543] /proc/taskinfo created
[ 3840.246843] [drm:vmw_stdu_primary_plane_atomic_update [vmwgfx]] *ERROR* Failed to update STDU.
orange@ubuntu:~/Desktop/OS$ cat /proc/taskinfo
Command: systemd
PID: 1
State: S
orange@ubuntu:~/Desktop/OS$ S
```