National Taiwan Normal University
Department of Computer Science and Information Engineering
CSC0016, Assignment 2

1 Information

1. The assignment is worth 100 points.
2. Individual work
3. Due at 12:00 11/29, i.e., Sunday noon
4. Deliver your assignment as a report in English or Chinese MS Word or PDF, whose maximum size is 6 pages without the reference and appendix.

2 Content

Your report shall contain the following directions at least.

- Hardware and software environment. Skip this part if it is identical to the previous assignment. Otherwise, describe it clearly.
- Implementation steps. Sequentially record steps and affected files, e.g., created or modified files or codes, which make the system calls work. Briefly describe the goal of each step.
- Demonstration. Based on the functions of system calls, show the execution results.
- Comments. At least one most challenging part and your solution, your comments, etc.
- Any reference if you use.
- Put full codes in appendix with tighter format.

1. (30 points) Add a simple system call to Linux kernel. The function of the system call is to print a message with the highest priority, `KERN_EMERG`, into the kernel log. The message must include your school id shown on the course website. Demonstrate the system call by calling it in user level and observing the results in kernel level, i.e., kernel log. Therefore, you have to write a test program in user level as well. Show partial kernel log, at least 5 lines, including the target message as a screen-shot.
2. (70 points) Add a system call to Linux kernel for collecting process information. The function of the system call is to collect the following information of the current process. Demonstrate the system call by calling it in user level under two different shells and show the collected information as screen-shots. Therefore, you have to write a test program in user level as well, and run the program under two different shells.

```
struct prinfo {
  long state; /* current state of process */
  long nice; /* process nice value */
  pid_t pid; /* process id */
  pid_t parent_pid; /* process id of parent */
  pid_t youngest_child_pid; /* pid of youngest child */
  unsigned long start_time; /* process start time */
  long user_time; /* CPU time spent in user mode */
  long sys_time; /* CPU time spent in system mode */
  long uid; /* user id of process owner */
  char comm[16]; /* name of program executed */
};
```

3 Tips

1. The implementations of `getuid()` and `getpid()` system calls in `kernel/timer.c` can provide some guidance.
2. Almost all of the information you need to fill in the `prinfo` structure can be found in the `task_struct` structure, defined in `include/linux/sched.h`.
3. Kernel file `include/asm/current.h` defines an inline function that returns the address of the `task_struct` of the current process.
4. Every system call must check the validity of the arguments passed by a caller. Linux kernel provides two functions, `copy_from_user()` and `copy_to_user()`, that not only check the validity but also transfer information between kernel and user levels. Use them to move information.