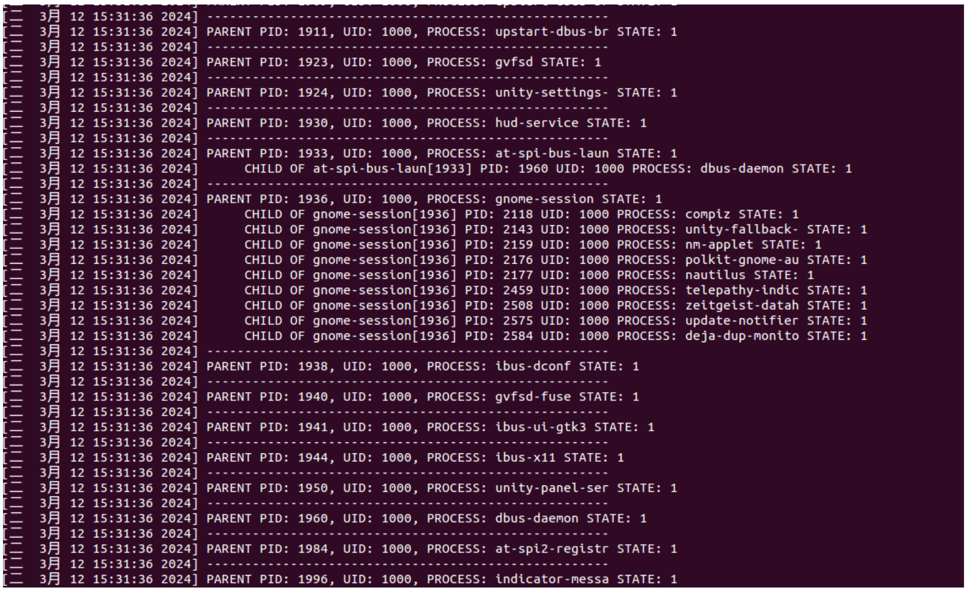
1. Module輸出　Command dmesg –T

(TIMESTAMP / PARENT PID: / UID: / PROCESS: / STATE:)

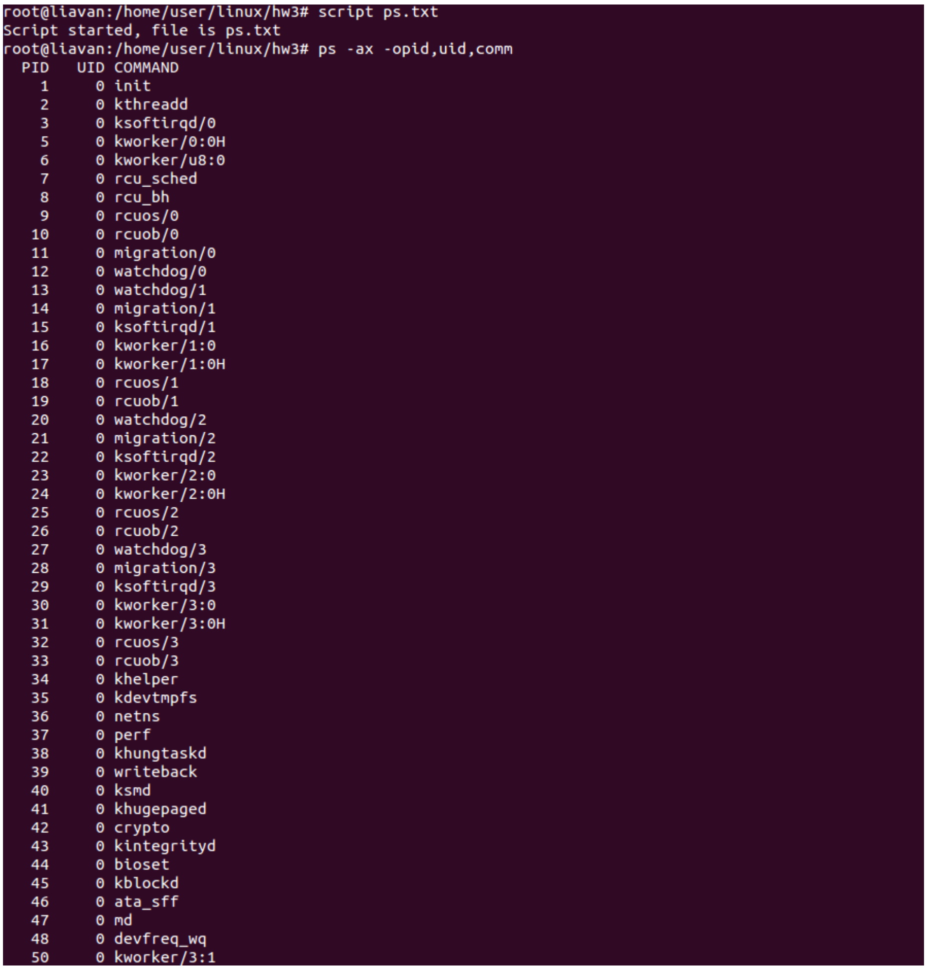


```
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1911, UID: 1000, PROCESS: upstart-dbus-br STATE: 1
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1923, UID: 1000, PROCESS: gvfsd STATE: 1
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1924, UID: 1000, PROCESS: unity-settings- STATE: 1
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1930, UID: 1000, PROCESS: hud-service STATE: 1
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1933, UID: 1000, PROCESS: at-spi-bus-laun STATE: 1
[二  3月 12 15:31:36 2024]      CHILD OF at-spi-bus-laun[1933] PID: 1960 UID: 1000 PROCESS: dbus-daemon STATE: 1
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1936, UID: 1000, PROCESS: gnome-session STATE: 1
[二  3月 12 15:31:36 2024]      CHILD OF gnome-session[1936] PID: 2118 UID: 1000 PROCESS: compiz STATE: 1
[二  3月 12 15:31:36 2024]      CHILD OF gnome-session[1936] PID: 2143 UID: 1000 PROCESS: unity-fallback- STATE: 1
[二  3月 12 15:31:36 2024]      CHILD OF gnome-session[1936] PID: 2159 UID: 1000 PROCESS: nm-applet STATE: 1
[二  3月 12 15:31:36 2024]      CHILD OF gnome-session[1936] PID: 2176 UID: 1000 PROCESS: polkit-gnome-au STATE: 1
[二  3月 12 15:31:36 2024]      CHILD OF gnome-session[1936] PID: 2177 UID: 1000 PROCESS: nautilus STATE: 1
[二  3月 12 15:31:36 2024]      CHILD OF gnome-session[1936] PID: 2459 UID: 1000 PROCESS: telepathy-indic STATE: 1
[二  3月 12 15:31:36 2024]      CHILD OF gnome-session[1936] PID: 2508 UID: 1000 PROCESS: zeitgeist-datah STATE: 1
[二  3月 12 15:31:36 2024]      CHILD OF gnome-session[1936] PID: 2575 UID: 1000 PROCESS: update-notifier STATE: 1
[二  3月 12 15:31:36 2024]      CHILD OF gnome-session[1936] PID: 2584 UID: 1000 PROCESS: deja-dup-monito STATE: 1
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1938, UID: 1000, PROCESS: ibus-dconf STATE: 1
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1940, UID: 1000, PROCESS: gvfsd-fuse STATE: 1
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1941, UID: 1000, PROCESS: ibus-ui-gtk3 STATE: 1
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1944, UID: 1000, PROCESS: ibus-x11 STATE: 1
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1950, UID: 1000, PROCESS: unity-panel-ser STATE: 1
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1960, UID: 1000, PROCESS: dbus-daemon STATE: 1
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1984, UID: 1000, PROCESS: at-spi2-registr STATE: 1
[二  3月 12 15:31:36 2024] ---------------------------------------------------
[二  3月 12 15:31:36 2024] PARENT PID: 1996, UID: 1000, PROCESS: indicator-messa STATE: 1
```

(範例片段圖片 完整在dmesg.txt文件中)

2. Command ps -ax -opid,uid,comm

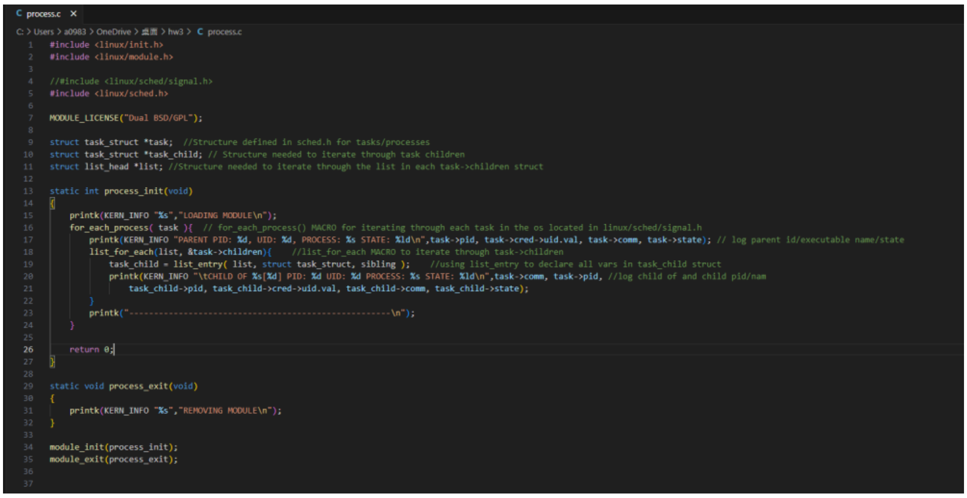
(PID / UID / COMMAND)



```
root@liavan:/home/user/linux/hw3# script ps.txt
Script started, file is ps.txt
root@liavan:/home/user/linux/hw3# ps -ax -opid,uid,comm
  PID   UID COMMAND
    1     0 init
    2     0 kthreadd
    3     0 ksoftirqd/0
    5     0 kworker/0:0H
    6     0 kworker/u8:0
    7     0 rcu_sched
    8     0 rcu_bh
    9     0 rcuos/0
   10     0 rcuob/0
   11     0 migration/0
   12     0 watchdog/0
   13     0 watchdog/1
   14     0 migration/1
   15     0 ksoftirqd/1
   16     0 kworker/1:0
   17     0 kworker/1:0H
   18     0 rcuos/1
   19     0 rcuob/1
   20     0 watchdog/2
   21     0 migration/2
   22     0 ksoftirqd/2
   23     0 kworker/2:0
   24     0 kworker/2:0H
   25     0 rcuos/2
   26     0 rcuob/2
   27     0 watchdog/3
   28     0 migration/3
   29     0 ksoftirqd/3
   30     0 kworker/3:0
   31     0 kworker/3:0H
   32     0 rcuos/3
   33     0 rcuob/3
   34     0 khelper
   35     0 kdevtmpfs
   36     0 netns
   37     0 perf
   38     0 khungtaskd
   39     0 writeback
   40     0 ksmd
   41     0 khugepaged
   42     0 crypto
   43     0 kintegrityd
   44     0 bioset
   45     0 kblockd
   46     0 ata_sff
   47     0 md
   48     0 devfreq_wq
   50     0 kworker/3:1
```

(範例片段圖片 完整在ps.txt文件中)

3. Source Code



```c
#include <linux/init.h>
#include <linux/module.h>

//#include <linux/sched/signal.h>
#include <linux/sched.h>

MODULE_LICENSE("Dual BSD/GPL");

struct task_struct *task;  //Structure defined in sched.h for tasks/processes
struct task_struct *task_child; // Structure needed to iterate through task children
struct list_head *list; //Structure needed to iterate through the list in each task->children struct

static int process_init(void)
{
    printk(KERN_INFO "%s","LOADING MODULE\n");
    for_each_process( task ){  // for_each_process() MACRO for iterating through each task in the os located in linux/sched/signal.h
        printk(KERN_INFO "PARENT PID: %d, UID: %d, PROCESS: %s STATE: %ld\n",task->pid, task->cred->uid.val, task->comm, task->state); // log parent id/executable name/state
        list_for_each(list, &task->children){    //list_for_each MACRO to iterate through task->children
            task_child = list_entry( list, struct task_struct, sibling );    //using list_entry to declare all vars in task_child struct
            printk(KERN_INFO "\tCHILD OF %s[%d] PID: %d UID: %d PROCESS: %s STATE: %ld\n",task->comm, task->pid, //log child of and child pid/nam
                task_child->pid, task_child->cred->uid.val, task_child->comm, task_child->state);
        }
        printk("-----------------------------------------------------\n");
    }

    return 0;
}

static void process_exit(void)
{
    printk(KERN_INFO "%s","REMOVING MODULE\n");
}

module_init(process_init);
module_exit(process_exit);


```