实验一 Linux内核模块编程

一、设计目的和内容要求

1. 设计目的
   Linux 提供的模块机制能动态扩充 Linux功能而无须重新编译内核，已经广泛应用在Linux内核的许多功能的实现中。在本实验中将学习模块的基本概念、原理及实现技术，然后利用内核模块编程访问进程的基本信息，加深对进程概念的理解，掌握基本的模块编程技术。
2. 内容要求
   1. 设计一个模块，要求列出系统中所有内核线程的程序名、PID、进程状态、进程优先级、父进程的PID。
   2. 设计一个带参数的模块，其参数为某个进程的-PID号，模块的功能是列出该进程的家族信息，包括父进程、兄弟进程和子进程的程序名、PID号及进程状态。
   3. 请根据自身情况，进一步阅读分析程序中用到的相关内核函数的源码实现。
3. 开发平台
   Ubuntu 24.04 华为ECS弹性云服务器

前置工作

```
$ mkdir os_1 #新建文件夹
$ cd os_1
$ mkdir tasks tasks_Addition
```

模块一（无参类型）

要求：列出系统中所有内核线程的程序名，PID，进程状态和进程优先级，父进程的PID。

```
#include <linux/init.h>
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/sched.h>
#include <linux/init_task.h>

static int all_task_init(void)
{
    struct task_struct *p;

    printk(KERN_INFO "Listing kernel threads:\n");
    printk(KERN_INFO "COMMAND\tPID\tSTATE\tPRIO\tPPID\n");

    for_each_process(p) {
        if (p->mm == NULL) {  // 内核线程检测
            // 安全获取父进程PID
            pid_t ppid = p->parent ? p->parent->pid : 0;

            printk(KERN_INFO "%-16s\t%-6d\t%-4ld\t%-4d\t%-6d\n",
                   p->comm,
                   p->pid,
                   p->__state,
                   p->prio,
                   ppid);
        }
    }
    return 0;
}

static void all_task_exit(void)
{
```

```
    printk(KERN_INFO "all_task module unloaded\n");
}

module_init(all_task_init);
module_exit(all_task_exit);

MODULE_LICENSE("GPL");
MODULE_AUTHOR("zxyhiatus");
MODULE_DESCRIPTION("Kernel thread lister module");
MODULE_VERSION("0.1");
```

对应的Makefile文件

```
obj-m := tasks.o
KDIR := /lib/modules/$(shell uname -r)/build
PWD := $(shell pwd)
default:
        make -C $(KDIR) M=$(PWD) modules
clean:
        make -C $(KDIR) M=$(PWD) clean
~
```

编译模块

```
$ make
```

```
root@os-practice:/usr/src/os_1/tasks# make
make -C /lib/modules/6.8.0-49-generic/build M=/usr/src/os_1/tasks modules
make[1]: Entering directory '/usr/src/linux-headers-6.8.0-49-generic'
warning: the compiler differs from the one used to build the kernel
  The kernel was built by: x86_64-linux-gnu-gcc-13 (Ubuntu 13.2.0-23ubuntu4) 13.2.0
  You are using:           gcc-13 (Ubuntu 13.2.0-23ubuntu4) 13.2.0
  CC [M]  /usr/src/os_1/tasks/tasks.o
In file included from ./include/asm-generic/bug.h:22,
                 from ./arch/x86/include/asm/bug.h:87,
                 from ./include/linux/bug.h:5,
                 from ./arch/x86/include/asm/paravirt.h:19,
                 from ./arch/x86/include/asm/cpuid.h:62,
                 from ./arch/x86/include/asm/processor.h:19,
                 from ./arch/x86/include/asm/timex.h:5,
                 from ./include/linux/timex.h:67,
                 from ./include/linux/time32.h:13,
                 from ./include/linux/time.h:60,
                 from ./include/linux/stat.h:19,
                 from ./include/linux/module.h:13,
                 from /usr/src/os_1/tasks/tasks.c:2:
/usr/src/os_1/tasks/tasks.c: In function 'all_task_init':
./include/linux/kern_levels.h:5:25: warning: format '%ld' expects argument of type 'long int', but argument 4 has type 'unsigned int' [-Wformat=]
    5 | #define KERN_SOH        "\001"          /* ASCII Start Of Header */
      |                         ^~~~~
./include/linux/printk.h:429:25: note: in definition of macro 'printk_index_wrap'
  429 |                 _p_func(_fmt, ##__VA_ARGS__);                           \
      |                         ^~~~
/usr/src/os_1/tasks/tasks.c:19:13: note: in expansion of macro 'printk'
   19 |             printk(KERN_INFO "%-16s\t%-6d\t%-4ld\t%-4d\t%-6d\n",
      |             ^~~~~~
./include/linux/kern_levels.h:14:25: note: in expansion of macro 'KERN_SOH'
   14 | #define KERN_INFO       KERN_SOH "6"    /* informational */
      |                         ^~~~~~~~
/usr/src/os_1/tasks/tasks.c:19:20: note: in expansion of macro 'KERN_INFO'
   19 |             printk(KERN_INFO "%-16s\t%-6d\t%-4ld\t%-4d\t%-6d\n",
      |                    ^~~~~~~~~
  MODPOST /usr/src/os_1/tasks/Module.symvers
  LD [M]  /usr/src/os_1/tasks/tasks.ko
  BTF [M] /usr/src/os_1/tasks/tasks.ko
Skipping BTF generation for /usr/src/os_1/tasks/tasks.ko due to unavailability of vmlinux
make[1]: Leaving directory '/usr/src/linux-headers-6.8.0-49-generic'
root@os-practice:/usr/src/os_1/tasks#
```

图一 编译无参文件

加载模块

```
$ insmod tasks.ko
$ modinfo tasks.ko #查看模块详细信息
```

```
root@os-practice:/usr/src/os_1/tasks# insmod tasks.ko
root@os-practice:/usr/src/os_1/tasks# modinfo tasks.ko
filename:       /usr/src/os_1/tasks/tasks.ko
version:        0.1
description:    Kernel thread lister module
author:         zxyhiatus
license:        GPL
srcversion:     83C3953DB9EE7F111DF57F4
depends:
retpoline:      Y
name:           tasks
vermagic:       6.8.0-49-generic SMP preempt mod_unload modversions
```

图二 加载模块并查看模块详细信息

查看结果

```
$ dmesg
```

```
[  960.467420] Listing kernel threads:
[  960.467421] COMMAND  PID     STATE   PRIO    PPID
[  960.467422] kthreadd         2       1       120     0
[  960.467425] pool_workqueue_  3       1       120     2
[  960.467427] kworker/R-rcu_g  4       1026    100     2
[  960.467429] kworker/R-rcu_p  5       1026    100     2
[  960.467430] kworker/R-slub_  6       1026    100     2
[  960.467432] kworker/R-netns  7       1026    100     2
[  960.467433] kworker/0:0      8       1026    120     2
[  960.467435] kworker/0:0H     9       1026    100     2
[  960.467436] kworker/u16:0    11      1026    120     2
[  960.467438] kworker/R-mm_pe  12      1026    100     2
[  960.467439] rcu_tasks_kthre  13      1026    120     2
[  960.467441] rcu_tasks_rude_  14      1026    120     2
[  960.467442] rcu_tasks_trace  15      1026    120     2
[  960.467443] ksoftirqd/0      16      1       120     2
[  960.467445] rcu_preempt      17      1026    120     2
[  960.467446] migration/0      18      1       0       2
[  960.467448] idle_inject/0    19      1       49      2
[  960.467449] cpuhp/0          20      1       120     2
[  960.467451] cpuhp/2          21      1       120     2
[  960.467452] idle_inject/2    22      1       49      2
[  960.467453] migration/2      23      1       0       2
[  960.467455] ksoftirqd/2      24      1       120     2
[  960.467456] kworker/2:0H     26      1026    100     2
[  960.467458] cpuhp/4          27      1       120     2
[  960.467459] idle_inject/4    28      1       49      2
[  960.467460] migration/4      29      1       0       2
[  960.467462] ksoftirqd/4      30      1       120     2
[  960.467463] kworker/4:0      31      1026    120     2
[  960.467465] kworker/4:0H     32      1026    100     2
[  960.467466] cpuhp/6          33      1       120     2
[  960.467468] idle_inject/6    34      1       49      2
[  960.467469] migration/6      35      1       0       2
[  960.467471] ksoftirqd/6      36      1       120     2
[  960.467472] kworker/6:0H     38      1026    100     2
[  960.467474] cpuhp/1          39      1       120     2
[  960.467476] idle_inject/1    40      1       49      2
[  960.467477] migration/1      41      1       0       2
[  960.467478] ksoftirqd/1      42      1       120     2
[  960.467480] kworker/1:0H     44      1026    100     2
[  960.467481] cpuhp/3          45      1       120     2
[  960.467482] idle_inject/3    46      1       49      2
[  960.467484] migration/3      47      1       0       2
[  960.467485] ksoftirqd/3      48      1       120     2
[  960.467486] kworker/3:0H     50      1026    100     2
[  960.467488] cpuhp/5          51      1       120     2
[  960.467489] idle_inject/5    52      1       49      2
[  960.467491] migration/5      53      1       0       2
[  960.467492] ksoftirqd/5      54      1       120     2
[  960.467493] kworker/5:0      55      1026    120     2
[  960.467494] kworker/5:0H     56      1026    100     2
[  960.467496] cpuhp/7          57      1       120     2
[  960.467497] idle_inject/7    58      1       49      2
[  960.467498] migration/7      59      1       0       2
[  960.467500] ksoftirqd/7      60      1       120     2
[  960.467501] kworker/7:0H     62      1026    100     2
[  960.467503] kdevtmpfs        63      1       120     2
[  960.467504] kworker/R-inet_  64      1026    100     2
[  960.467505] kworker/u16:1    65      1026    120     2
[  960.467507] kauditd          66      8193    120     2
[  960.467508] khungtaskd       67      1       120     2
```

```
[  960.467510] oom_reaper       68      8193    120     2
[  960.467511] kworker/u16:2    69      1026    120     2
[  960.467513] kworker/R-write  70      1026    100     2
[  960.467514] kcompactd0       71      8193    120     2
[  960.467515] ksmd             72      8193    125     2
[  960.467517] khugepaged       73      8193    139     2
[  960.467518] kworker/R-kinte  74      1026    100     2
[  960.467520] kworker/R-kbloc  75      1026    100     2
[  960.467521] kworker/R-blkcg  76      1026    100     2
[  960.467523] irq/9-acpi       77      1       49      2
[  960.467524] kworker/1:1      78      1026    120     2
[  960.467525] kworker/2:1      79      1026    120     2
[  960.467527] kworker/R-tpm_d  80      1026    100     2
[  960.467528] kworker/R-ata_s  81      1026    100     2
[  960.467530] kworker/R-md     82      1026    100     2
[  960.467531] kworker/R-md_bi  83      1026    100     2
[  960.467532] kworker/R-edac-  84      1026    100     2
[  960.467534] kworker/R-devfr  85      1026    100     2
[  960.467535] watchdogd        86      1       49      2
[  960.467536] kworker/0:1H     87      1026    100     2
[  960.467538] kworker/6:1      88      1026    120     2
[  960.467540] kswapd0          89      1       120     2
[  960.467541] ecryptfs-kthrea  90      8193    120     2
[  960.467542] kworker/R-kthro  91      1026    100     2
[  960.467544] kworker/7:1      92      1026    120     2
[  960.467545] kworker/R-acpi_  93      1026    100     2
[  960.467546] scsi_eh_0        95      1       120     2
[  960.467548] kworker/R-scsi_  96      1026    100     2
[  960.467549] scsi_eh_1        97      1       120     2
[  960.467550] kworker/R-scsi_  98      1026    100     2
[  960.467552] kworker/u16:3    99      1026    120     2
[  960.467553] kworker/R-mld    100     1026    100     2
[  960.467555] kworker/1:1H     101     1026    100     2
[  960.467556] kworker/R-ipv6_  102     1026    100     2
[  960.467558] kworker/R-kstrp  109     1026    100     2
[  960.467559] kworker/4:1      110     1026    120     2
[  960.467561] kworker/u17:0    113     1026    100     2
[  960.467562] kworker/R-crypt  118     1026    100     2
[  960.467564] kworker/u16:4    119     1026    120     2
[  960.467565] kworker/7:2      120     1026    120     2
[  960.467567] kworker/2:2      121     1026    120     2
[  960.467568] kworker/R-charg  131     1026    100     2
[  960.467569] kworker/4:1H     155     1026    100     2
[  960.467571] kworker/3:1H     159     1026    100     2
[  960.467572] kworker/6:1H     170     1026    100     2
[  960.467573] kworker/2:1H     180     1026    100     2
[  960.467575] kworker/5:1H     181     1026    100     2
[  960.467576] kworker/7:1H     182     1026    100     2
[  960.467577] kworker/1:2      188     1026    120     2
[  960.467579] kworker/R-raid5  238     1026    100     2
[  960.467580] jbd2/vda1-8      278     1       120     2
[  960.467581] kworker/R-ext4-  279     1026    100     2
[  960.467583] kworker/5:2      346     1026    120     2
[  960.467584] kworker/R-kmpat  377     1026    100     2
[  960.467586] kworker/R-kmpat  378     1026    100     2
[  960.467587] kworker/0:2      379     1026    120     2
[  960.467588] kworker/3:6      397     1026    120     2
[  960.467590] kworker/3:7      398     1026    120     2
[  960.467592] psimon           603     1       98      2
[  960.467594] kworker/R-nfit   896     1026    100     2
[  960.467598] kworker/6:3      1530    1026    120     2
[  960.467601] kworker/u16:5    1664    1026    120     2
```

```
[  960.467602] kworker/u16:6    1675    1026    120     2
[  960.467604] kworker/u16:7    1719    1026    120     2
[  960.467605] psimon           1767    1       98      2
[  960.467607] psimon           1776    1       98      2
[  960.467609] kworker/R-tls-s  1905    1026    100     2
[  960.467611] kworker/1:0      2584    1026    120     2
[  960.467612] kworker/1:3      2585    1026    120     2
root@os-practice:/usr/src/os_1/tasks#
```

图三 模块执行结果

卸载模块

```
$ rmmod tasks.ko
$ dmesg  #查看是否卸载成功
```

```
[ 1165.063327] all_task module unloaded
```

图四 模块卸载输出

检验结果

```
$ ps aux #列出所有线程/进程
```

```
root@os-practice:/usr/src/os_1/tasks# ps aux
USER         PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  1.0  0.0  22684 13416 ?        Ss   10:11   0:01 /sbin/init noibrs
root           2  0.0  0.0      0     0 ?        S    10:11   0:00 [kthreadd]
root           3  0.0  0.0      0     0 ?        S    10:11   0:00 [pool_workqueue_release]
root           4  0.0  0.0      0     0 ?        I<   10:11   0:00 [kworker/R-rcu_g]
root           5  0.0  0.0      0     0 ?        I<   10:11   0:00 [kworker/R-rcu_p]
root           6  0.0  0.0      0     0 ?        I<   10:11   0:00 [kworker/R-slub_]
root           7  0.0  0.0      0     0 ?        I<   10:11   0:00 [kworker/R-netns]
root           8  0.0  0.0      0     0 ?        I    10:11   0:00 [kworker/0:0-mm_percpu_wq]
root           9  0.0  0.0      0     0 ?        I<   10:11   0:00 [kworker/0:0H-events_highpri]
root          10  0.0  0.0      0     0 ?        I    10:11   0:00 [kworker/0:1-rcu_gp]
root          11  0.0  0.0      0     0 ?        I    10:11   0:00 [kworker/u16:0-events_unbound]
root          12  0.0  0.0      0     0 ?        I<   10:11   0:00 [kworker/R-mm_pe]
root          13  0.0  0.0      0     0 ?        I    10:11   0:00 [rcu_tasks_kthread]
root          14  0.0  0.0      0     0 ?        I    10:11   0:00 [rcu_tasks_rude_kthread]
root          15  0.0  0.0      0     0 ?        I    10:11   0:00 [rcu_tasks_trace_kthread]
root          16  0.0  0.0      0     0 ?        S    10:11   0:00 [ksoftirqd/0]
root          17  0.0  0.0      0     0 ?        I    10:11   0:00 [rcu_preempt]
root          18  0.0  0.0      0     0 ?        S    10:11   0:00 [migration/0]
root          19  0.0  0.0      0     0 ?        S    10:11   0:00 [idle_inject/0]
root          20  0.0  0.0      0     0 ?        S    10:11   0:00 [cpuhp/0]
root          21  0.0  0.0      0     0 ?        S    10:11   0:00 [cpuhp/2]
root          22  0.0  0.0      0     0 ?        S    10:11   0:00 [idle_inject/2]
root          23  0.0  0.0      0     0 ?        S    10:11   0:00 [migration/2]
root          24  0.0  0.0      0     0 ?        S    10:11   0:00 [ksoftirqd/2]
root          25  0.0  0.0      0     0 ?        I    10:11   0:00 [kworker/2:0-cgroup_destroy]
root          26  0.0  0.0      0     0 ?        I<   10:11   0:00 [kworker/2:0H-kblockd]
root          27  0.0  0.0      0     0 ?        S    10:11   0:00 [cpuhp/4]
root          28  0.0  0.0      0     0 ?        S    10:11   0:00 [idle_inject/4]
root          29  0.0  0.0      0     0 ?        S    10:11   0:00 [migration/4]
root          30  0.0  0.0      0     0 ?        S    10:11   0:00 [ksoftirqd/4]
root          31  0.0  0.0      0     0 ?        I    10:11   0:00 [kworker/4:0-events]
root          32  0.0  0.0      0     0 ?        I<   10:11   0:00 [kworker/4:0H-events_highpri]
root          33  0.0  0.0      0     0 ?        S    10:11   0:00 [cpuhp/6]
root          34  0.0  0.0      0     0 ?        S    10:11   0:00 [idle_inject/6]
root          35  0.0  0.0      0     0 ?        S    10:11   0:00 [migration/6]
root          36  0.0  0.0      0     0 ?        S    10:11   0:00 [ksoftirqd/6]
root          37  0.0  0.0      0     0 ?        I    10:11   0:00 [kworker/6:0-rcu_gp]
root          38  0.0  0.0      0     0 ?        I<   10:11   0:00 [kworker/6:0H-events_highpri]
root          39  0.0  0.0      0     0 ?        S    10:11   0:00 [cpuhp/1]
root          40  0.0  0.0      0     0 ?        S    10:11   0:00 [idle_inject/1]
root          41  0.0  0.0      0     0 ?        S    10:11   0:00 [migration/1]
root          42  0.0  0.0      0     0 ?        S    10:11   0:00 [ksoftirqd/1]
root          43  0.0  0.0      0     0 ?        I    10:11   0:00 [kworker/1:0-slub_flushwq]
root          44  0.0  0.0      0     0 ?        I<   10:11   0:00 [kworker/1:0H-events_highpri]
root          45  0.0  0.0      0     0 ?        S    10:11   0:00 [cpuhp/3]
```

图五 检验结果

模块二（有参）

要求:
其参数为某个进程的-PID号，模块的功能是列出该进程的家族信息，包括父进程、兄弟进程和子进程的程序名、PID号及进程状态。

```c
#include <linux/init.h>
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/sched.h>
#include <linux/init_task.h>
#include <linux/moduleparam.h>

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

```
MODULE_DESCRIPTION("Process Family Tree Viewer");
MODULE_VERSION("1.0");

static int pid = 1;  // 默认查看 init 进程
module_param(pid, int, 0644);

static int family_task_init(void)
{
    struct task_struct *target = NULL;
    struct task_struct *parent = NULL;
    int found = 0;

    printk(KERN_INFO "=== Process Family Tree for PID: %d ===\n", pid);

    // 查找目标进程
    for_each_process(target) {
        if (target->pid == pid) {
            found = 1;
            break;
        }
    }

    if (!found) {
        printk(KERN_WARNING "Process with PID %d not found!\n", pid);
        return -ESRCH;
    }

    // 打印目标进程信息
    printk(KERN_INFO "[Target Process]\n");
    printk(KERN_INFO "%-16s %-6s %-6s\n", "COMMAND", "PID", "STATE");
    printk(KERN_INFO "%-16s %-6d %-6ld\n",
           target->comm, target->pid, target->__state);

    // 处理父进程
    parent = target->parent;
    printk(KERN_INFO "\n[Parent Process]\n");
    if (parent) {
        printk(KERN_INFO "%-16s %-6d %-6ld\n",
               parent->comm, parent->pid, parent->__state);
    } else {
        printk(KERN_INFO "No parent (kernel thread)\n");
    }

    // 处理子进程
    printk(KERN_INFO "\n[Children Processes]\n");
    if (!list_empty(&target->children)) {
        struct task_struct *child;
        list_for_each_entry(child, &target->children, sibling) {
            printk(KERN_INFO "%-16s %-6d %-6ld\n",
                   child->comm, child->pid, child->__state);
        }
    } else {
        printk(KERN_INFO "No children\n");
    }

    // 处理兄弟进程
    printk(KERN_INFO "\n[Sibling Processes]\n");
    if (parent && !list_empty(&parent->children)) {
        struct task_struct *sibling;
```

```
        list_for_each_entry(sibling, &parent->children, sibling) {
            if (sibling->pid != pid) {
                printk(KERN_INFO "%-16s %-6d %-6ld\n",
                       sibling->comm, sibling->pid, sibling->__state);
            }
        }
    } else {
        printk(KERN_INFO "No siblings\n");
    }

    return 0;
}

static void family_task_exit(void)
{
    printk(KERN_INFO "tasks_Addition module unloaded\n");
}

module_init(family_task_init);
module_exit(family_task_exit);
```

编辑Makefile

```
obj-m := tasks_Addition.o
KDIR := /lib/modules/$(shell uname -r)/build
PWD := $(shell pwd)
default:
        make -C $(KDIR) M=$(PWD) modules
clean:
        make -C $(KDIR) M=$(PWD) clean
```

编译模块

```
  MODPOST /usr/src/os_1/tasks_Addition/Module.symvers
  LD [M]  /usr/src/os_1/tasks_Addition/tasks_Addition.ko
  BTF [M] /usr/src/os_1/tasks_Addition/tasks_Addition.ko
Skipping BTF generation for /usr/src/os_1/tasks_Addition/tasks_Addition.ko due to unavailability of vmlinux
make[1]: Leaving directory '/usr/src/linux-headers-6.8.0-49-generic'
```

加载模块&卸载模块

```
[ 1714.779694] === Process Family Tree for PID: 1 ===
[ 1714.779708] [Target Process]
[ 1714.779709] COMMAND          PID    STATE
[ 1714.779710] systemd          1      1
[ 1714.779712]
               [Parent Process]
[ 1714.779713] swapper/0        0      0
[ 1714.779714]
               [Children Processes]
[ 1714.779715] systemd-journal  352    1
[ 1714.779717] multipathd       409    8193
[ 1714.779719] systemd-udevd    431    1
[ 1714.779720] systemd-resolve  662    1
[ 1714.779721] uniagent         911    8193
[ 1714.779722] dbus-daemon      912    1
[ 1714.779724] irqbalance       917    1
[ 1714.779725] polkitd          919    1
[ 1714.779726] systemd-logind   931    1
[ 1714.779728] udisksd          933    1
[ 1714.779729] NetworkManager   940    1
[ 1714.779730] wpa_supplicant   947    1
[ 1714.779731] ModemManager     999    1
[ 1714.779732] rsyslogd         1001   1
[ 1714.779734] hostwatch        1018   8193
[ 1714.779735] hostguard        1071   1
[ 1714.779736] unattended-upgr  1191   1
[ 1714.779738] uniagentd        1196   1
[ 1714.779739] chronyd          1205   1
[ 1714.779740] cron             1305   8193
[ 1714.779742] agetty           1507   1
[ 1714.779743] sshd             1510   1
[ 1714.779744] agetty           1512   1
[ 1714.779745] containerserver  1577   8193
[ 1714.779747] systemd          1769   1
[ 1714.779748] telescope        1922   8193
[ 1714.779750]
               [Sibling Processes]
[ 1714.779751] kthreadd         2      1
[ 1727.793976] tasks Addition module unloaded
```

Q:
1.如何判断出这个进程是内核级进程?
2.父子进程逻辑树

A:
1.通过tasks结构中的mm链表头，如果不为空，则为内核级进程
2.