实验报告

课程名称： 操作系统实验

学 院： 计算机科学与工程学院

专 业： 软件工程 **班级：** 2018级 班

姓 名： KAFLE SAMRAT **学号：** 201801060933

2020年12月20日

山东科技大学教务处制

实 验 报 告

组 别		**姓 名**	KAFLE SAMRAT	**同组实验者**	
实验项目名称	添加最简单的linux 内核模块			**实验日期**	月 日
教 师 评 语					
实验成绩:			**指导教师（签名）：** 2020年 月 日		

一. 实验目的

熟练掌握基本的Linux内核模块开发框架和编译方法。

熟练掌握Linux内核模块添加流程。

理解Linux内核模块代码中的一些常见宏和参数。

掌握Linux内核模块程序和应用程序的差异。

深入理解操作系统为用户提供服务的方式、方法

深入理解计算机程序的运行方式

二, 实验要求:

通过阅读、执行hello.c及其对应的Makefile文

件，理解Linux内核模块LKM的基本框架和运行方式、原理。

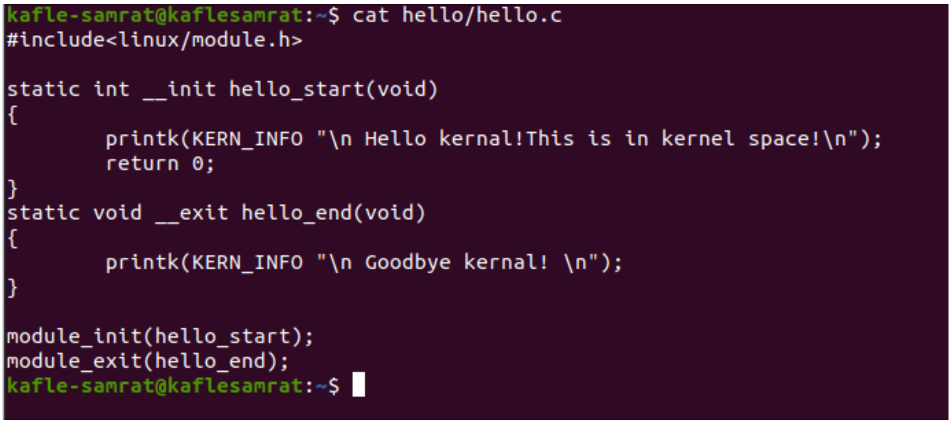
结合操作系统知识，通过实验深入理解计算机程序在操作系统支持下的运行方式。

三. 实验内容

从教材提供的电子资源中找到或者按教材提示自己编写简单的Linux内核模块kello.c及其对应的Makefile文件

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

static int __init hello_start(void)
{
  printk(KERN_INFO "\n Hello kernal!This is in
kernel space!\n");
```

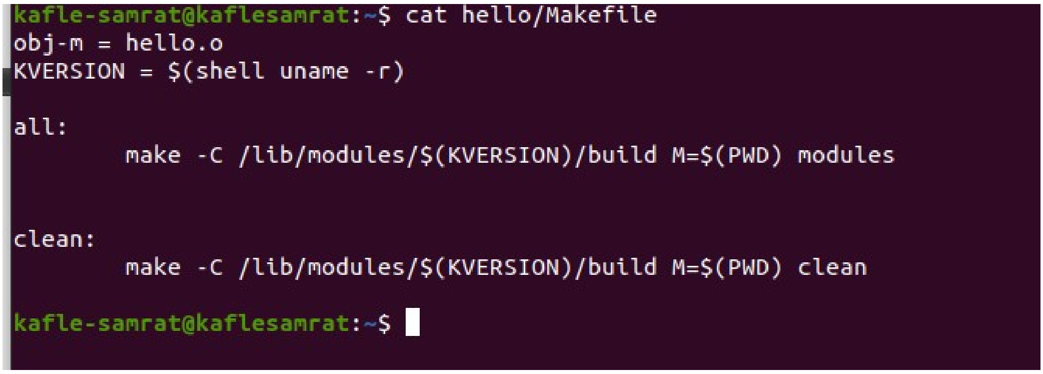
```
    return 0;
}
static void __exit hello_end(void)
{
    printk(KERN_INFO "\n Goodbye kernal! \n");
}

module_init(hello_start);
module_exit(hello_end);
```



```
kafle-samrat@kaflesamrat:~$ cat hello/hello.c
#include<linux/module.h>

static int __init hello_start(void)
{
        printk(KERN_INFO "\n Hello kernal!This is in kernel space!\n");
        return 0;
}
static void __exit hello_end(void)
{
        printk(KERN_INFO "\n Goodbye kernal! \n");
}

module_init(hello_start);
module_exit(hello_end);
kafle-samrat@kaflesamrat:~$ 
```

```
obj-m = hello.o
KVERSION = $(shell uname -r)
```

```
all:

   make -C /lib/modules/$(KVERSION)/build M=$(PWD)

modules



clean:

   make -C /lib/modules/$(KVERSION)/build M=$(PWD)

clean
```



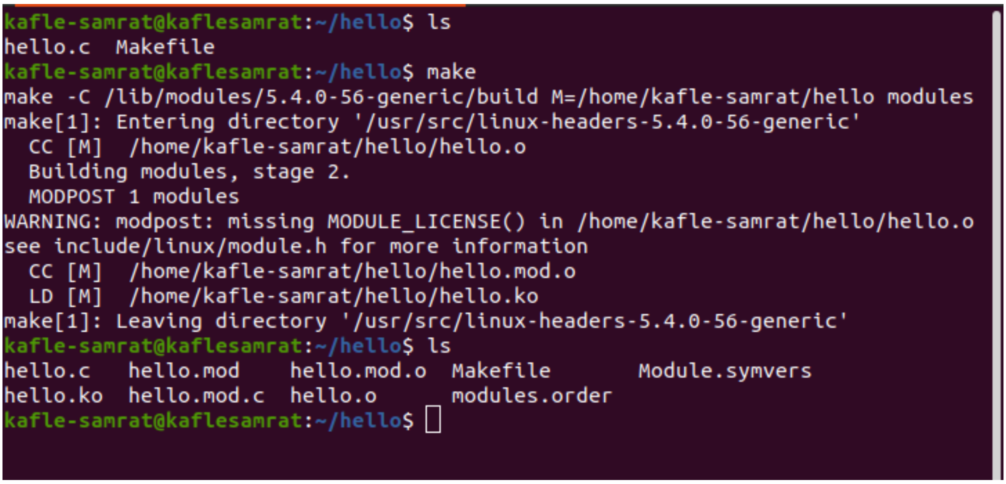
```
kafle-samrat@kaflesamrat:~$ cat hello/Makefile
obj-m = hello.o
KVERSION = $(shell uname -r)

all:
	make -C /lib/modules/$(KVERSION)/build M=$(PWD) modules


clean:
	make -C /lib/modules/$(KVERSION)/build M=$(PWD) clean

kafle-samrat@kaflesamrat:~$ 
```

编译、安装、删除该模块，查看该模块的安装位置、运行情况

本次采用单独编译、动态插入内核；把将开发的内核代码文件直接进行编译，然后使用命令动态插入内核或者从内核卸载。

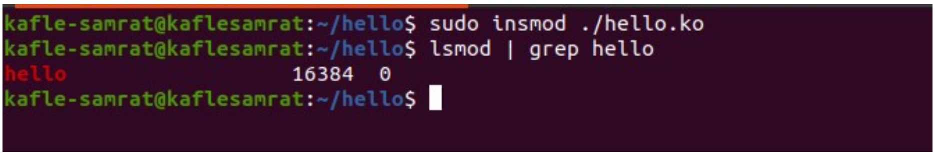
优点：编译速度快；单独调试代码

缺点：每次系统启动后都需要再加载代码



```
kafle-samrat@kaflesamrat:~/hello$ ls
hello.c  Makefile
kafle-samrat@kaflesamrat:~/hello$ make
make -C /lib/modules/5.4.0-56-generic/build M=/home/kafle-samrat/hello modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-56-generic'
  CC [M]  /home/kafle-samrat/hello/hello.o
  Building modules, stage 2.
  MODPOST 1 modules
WARNING: modpost: missing MODULE_LICENSE() in /home/kafle-samrat/hello/hello.o
see include/linux/module.h for more information
  CC [M]  /home/kafle-samrat/hello/hello.mod.o
  LD [M]  /home/kafle-samrat/hello/hello.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-56-generic'
kafle-samrat@kaflesamrat:~/hello$ ls
hello.c   hello.mod    hello.mod.o  Makefile      Module.symvers
hello.ko  hello.mod.c  hello.o      modules.order
kafle-samrat@kaflesamrat:~/hello$ 
```

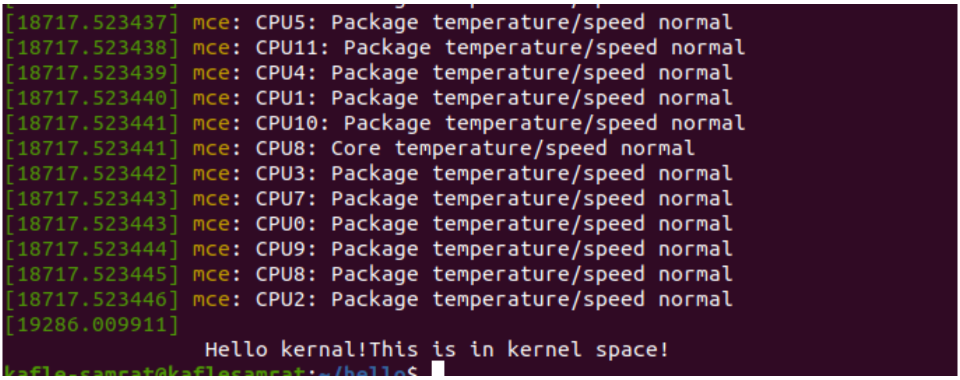
.



```
kafle-samrat@kaflesamrat:~/hello$ sudo insmod ./hello.ko
kafle-samrat@kaflesamrat:~/hello$ lsmod | grep hello
hello                  16384  0
kafle-samrat@kaflesamrat:~/hello$ 
```

dmesg 后显示：

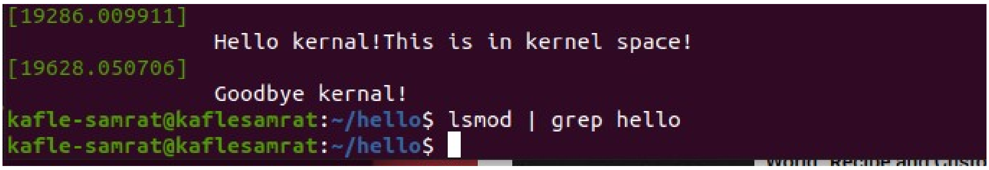
.



```
[18717.523437] mce: CPU5: Package temperature/speed normal
[18717.523438] mce: CPU11: Package temperature/speed normal
[18717.523439] mce: CPU4: Package temperature/speed normal
[18717.523440] mce: CPU1: Package temperature/speed normal
[18717.523441] mce: CPU10: Package temperature/speed normal
[18717.523441] mce: CPU8: Core temperature/speed normal
[18717.523442] mce: CPU3: Package temperature/speed normal
[18717.523443] mce: CPU7: Package temperature/speed normal
[18717.523443] mce: CPU0: Package temperature/speed normal
[18717.523444] mce: CPU9: Package temperature/speed normal
[18717.523445] mce: CPU8: Package temperature/speed normal
[18717.523446] mce: CPU2: Package temperature/speed normal
[19286.009911]
                Hello kernal!This is in kernel space!
kafle-samrat@kaflesamrat:~/hello$ 
```

删除模块：

sudo rmmod hello.ko



```
[19286.009911]
                Hello kernal!This is in kernel space!
[19628.050706]
                Goodbye kernal!
kafle-samrat@kaflesamrat:~/hello$ lsmod | grep hello
kafle-samrat@kaflesamrat:~/hello$ 
```

实验心得：

通过这次实验，我学习了基本的Linux内核模块开发框架和编译方法，熟悉了添加Linux内核模块的过程，了解了程序的工作原理。

实验报告

组别		姓名	KAFLE SAMRAT	同组实验者	
实验项目名称	进程间通信			实验日期	月 日
教师评语					
实验成绩:			指导教师（签名）： 2020 年 月 日		

一. 实验目的

掌握管道、信号、共享内存、消息队列等进程间通信机制;

二. 实验内容

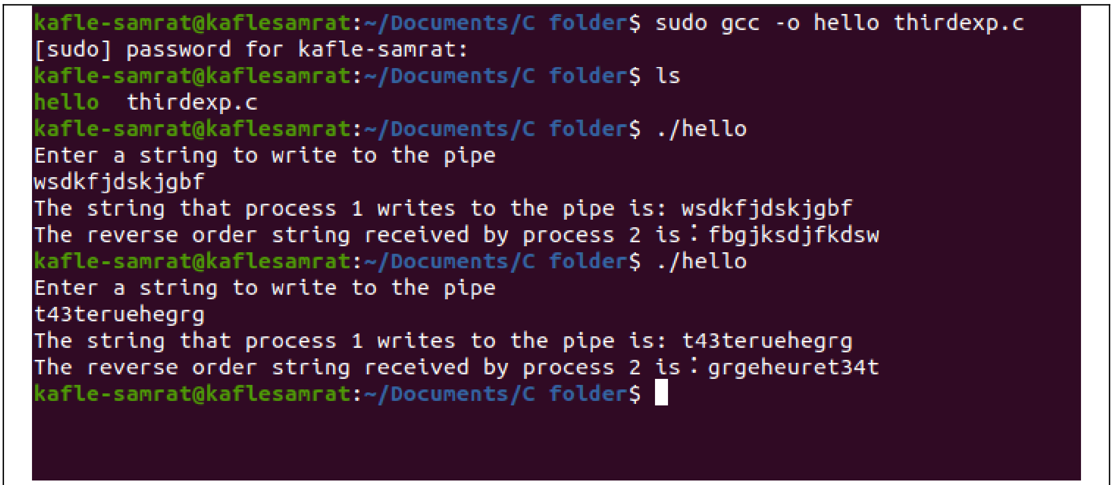
```
#include <stdio.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <fcntl.h>

#include<sys/wait.h>

#include <unistd.h>
```

```
#include <stdlib.h>
#include <string.h>
int main(){
    pid_t pid;
    int len;
    int pipe_fd[2];
    char str_read[100], str_write[100];
    char str[100];

    memset(str_read,0,sizeof(str_read));
    memset(str,0,sizeof(str));
    if(pipe(pipe_fd)<0){
        printf("Failed to create pipeline："); 
        return-1;
    }
    pid = fork();
    if(pid == 0){
        close(pipe_fd[1]);
```

```
            if((len = read(pipe_fd[0],str_read,100))>0){
            int i = 0;
            for(i = 0; i < len; i++){
                str[i] = str_read[len - i - 1];
            }
            printf("The reverse order string received by process
2 is：%s\n",str);
        }
        close(pipe_fd[0]);
        exit(0);
    }
    else if(pid > 0){
        close(pipe_fd[0]);
        printf("Enter a string to write to the pipe\n");
        scanf("%s", str_write) ;
        if(write(pipe_fd[1],str_write, strlen(str_write)) != -1){
            printf("The string that process 1 writes to the pipe
is: %s\n", str_write);
```

```
        }
        close(pipe_fd[1]);
        waitpid(pid,NULL,0);
        exit(0);
    }
    else if(pid < 0){
        printf("Child process creation failed");
        exit(0);
    }
}
```

实验结果：



```
kafle-samrat@kaflesamrat:~/Documents/C folder$ sudo gcc -o hello thirdexp.c
[sudo] password for kafle-samrat:
kafle-samrat@kaflesamrat:~/Documents/C folder$ ls
hello  thirdexp.c
kafle-samrat@kaflesamrat:~/Documents/C folder$ ./hello
Enter a string to write to the pipe
wsdkfjdskjgbf
The string that process 1 writes to the pipe is: wsdkfjdskjgbf
The reverse order string received by process 2 is : fbgjksdjfkdsw
kafle-samrat@kaflesamrat:~/Documents/C folder$ ./hello
Enter a string to write to the pipe
t43teruehegrg
The string that process 1 writes to the pipe is: t43teruehegrg
The reverse order string received by process 2 is : grgeheuret34t
kafle-samrat@kaflesamrat:~/Documents/C folder$ 
```

实验报告

组别		姓名	KAFLE SAMRAT	同组实验者	
实验项目名称	LINUX 内存管理			实验日期	月 日
教师评语					
实验成绩:			指导教师（签名）： 2020年 月 日		

一. 实验目的

1， 通过本次试验体会操作系统中内存的分配模式;

2， 掌握内存分配的方法（FF,BF,WF）；

3， 学会进程的建立，当一个进程被终止时内存是如何处理被释放块，并当内存不满足进程申请时是如何使用内存紧凑;

4， 掌握内存回收过程及实现方法;

5， 学会进行内存的申请释放和管理;

二. 实验内容

```
#include<stdio.h>
    #include<stdlib.h>



    #define PROCESS_NAME_LEN 32 /* Process name length */
#define MIN_SLICE 10          /* The size of the smallest fragment */
```

```
#define DEFAULT_MEM_SIZE 1024      /* Memory size */
#define DEFAULT_MEM_START 0              /* The starting position */

/* Memory allocation algorithm */
#define MA_FF 1
#define MA_BF 2
#define MA_WF 3



    /* Describes the data structure for each free block */
    typedef struct free_block_type{
        int size;
        int start_addr;
        struct free_block_type *next;
    }FBT;


    /* A description of the memory blocks allocated by each process */
    typedef struct allocated_block{
        int pid;
        int size;
        int start_addr;
        char process_name[PROCESS_NAME_LEN];
        struct allocated_block *next;
    }AB;


    /* A pointer to the first pointer to a list of free blocks in memory */
    FBT *free_block;


    /* The process allocates the first pointer to a linked list of memory blocks */
    AB *allocated_block_head = NULL;
    int mem_size = DEFAULT_MEM_SIZE; /* Memory size */
    int ma_algorithm = MA_FF; /* Current allocation algorithm */
    static int pid = 0; /*The initial pid*/
    int flag = 0; /* Sets the memory size flag */
    int min_mem_size = 10; /* Sets a flag that the remaining partitions are too small */


    FBT *init_free_block(int mem_size);
    void display_menu();
    int set_mem_size();
    int display_mem_usage();
    int dispose(AB *free_ab);
    int free_mem();
    int kill_process();
    int allocate_mem(AB *ab);
    int new_process();
    void rearrange_FF();
    void rearrange_BF();
    void rearrange_WF();
    void rearrange(int algorithm);
    void set_algorithm();


    void do_exit(){
        /* The general operating system will reclaim the applied memory after the program exit or return, so the place is empty.*/
        return;
    }


    int main(int argc, char const *argv[]){
        /* code */
        char choice;
        pid = 0;
        free_block = init_free_block(mem_size); // Initialize the free zone
        while(1){
```

```
        fflush(stdin);
        display_menu(); // According to the menu
        fflush(stdin);
        while((choice = getchar()) != '\n'){
        //choice = getchar();
        fflush(stdin);
        switch(choice){
            case '1':set_mem_size();break;
            case '2':set_algorithm();flag = 1;break;
            case '3':new_process();flag = 1;break;
            case '4':kill_process();flag = 1;break;
            case '5':display_mem_usage();flag = 1;break;
            case '0':do_exit();exit(0);
            default: break;
        }
        fflush(stdin);
        }
    }
}


void display_menu(){
    puts("");
    printf("1 - Set memory size(fedault=%d)\n",DEFAULT_MEM_SIZE);
    printf("2 - Select memory allocation algorithm\n");
    printf("3 - New process\n");
    printf("4 - Terminate a process \n");
    printf("5 - Display memory usage\n");
    printf("0 - Exit\n");
}


// Initializes a linked list of free partitions
FBT *init_free_block(int mem_size){
    FBT *fb;
    fb = (FBT*)malloc(sizeof(FBT));
    if(fb==NULL){
        printf("No mem\n");
        return NULL;
    }
    fb->size = mem_size;
    fb->start_addr = DEFAULT_MEM_START;
    fb->next = NULL;
    return fb;
}


// Reset the memory size
int set_mem_size(){
    int size;
    if(flag!=0){
        printf("Cannot set memory size again\n");
        return 0;
    }
    printf("Total memory size =");
    scanf("%d",&size);
    if(size>0){
        mem_size = size;
        free_block->size = mem_size;
    }
    flag = 1;
    min_mem_size = mem_size / 100;
    return 1;
}


int display_mem_usage(){
    /* Displays current memory usage, including free partitions and allocated ones */
    FBT *fbt = free_block;
    AB *ab = allocated_block_head;
```

```
        // if(fbt == NULL) return -1;
        printf("\e[0;31;1m------------------------------------------------------------------\e[0m\n");
        // Display free area
        printf("\e[0;32;1mFree Memory:\e[0m\n");
        printf("\e[0;33;1m%20s %20s\e[0m\n","     start_addr","       size");
        while(fbt!=NULL){
            printf("%20d %20d\n",fbt->start_addr,fbt->size);
            fbt = fbt->next;
        }


        // Displays the allocated areas
        printf("\n");
        printf("\e[0;35;1mUsed Memory:\e[0m\n");
        printf("\e[0;33;1m%10s %20s %20s %10s\e[0m\n","PID","ProcessName","start_addr","size");
        while(ab != NULL){
            printf("%10d %20s %20d %10d\n",ab->pid,ab->process_name,ab->start_addr,ab->size);
            ab = ab->next;
        }
        printf("\e[0;31;1m------------------------------------------------------------------\e[0m\n");
        return 0;
    }


    // Release the linked list nodes
    int dispose(AB *free_ab){
        /* Release ab data structure nodes */
        AB *pre,*ab;
        if(free_ab == allocated_block_head){
            // If you want to release the first node
            allocated_block_head = allocated_block_head->next;
            free(free_ab);
            return 1;
        }
        pre = allocated_block_head;
        ab = allocated_block_head->next;
        while(ab!=free_ab){
            pre = ab;
            ab = ab->next;
        }
        pre->next = ab->next;
        free(ab);
        return 2;
    }


    // Frees up memory occupied by the process
    int free_mem(AB *ab){
        /* Return the allocated areas represented by AB and make a possible merge */
        int algorithm = ma_algorithm;
        FBT *fbt,*pre,*work;
        fbt = (FBT*)malloc(sizeof(FBT));
        if(!fbt) return -1;
        /*
        For a possible merger, the basic strategy is as follows

1. Insert the newly released node to the end of the queue in the free partition

2. Organize free lists by address

3. Check and merge adjacent free partitions

4. Reorder the free linked list according to the current algorithm             */
        fbt->size = ab->size;
        fbt->start_addr = ab->start_addr;


        // Insert to the end
        work = free_block;
        if(work == NULL){
            free_block = fbt;
            fbt->next == NULL;
        }else{
            while(work ->next != NULL){
                work = work->next;
            }
            fbt->next = work->next;
            work->next = fbt;
        }
        // Rearrange the layout according to the address
        rearrange_FF();


        /* Merge possible partitions; If two free partitions are connected, they are merged*/
        pre = free_block;
        while(pre->next){
            work = pre->next;
            if(pre->start_addr + pre->size == work->start_addr ){
                pre->size = pre->size + work->size;
                pre->next = work->next;
                free(work);
                continue;
            }else{
                pre = pre->next;
            }
        }


        // Sort by the current algorithm
        rearrange(ma_algorithm);
        return 1;
    }


    // Find the linked list node corresponding to pid
```

```
AB *find_process(int pid){
    AB *tmp = allocated_block_head;
    while(tmp != NULL){
        if(tmp->pid == pid){
            return tmp;
        }
        tmp = tmp->next;
    }
    printf("\e[0;31;1m Cannot find pid:%d \e[0m\n",pid);
    return NULL;
}


int kill_process(){
    AB *ab;
    int pid;
    printf("Kill Process,pid=");
    scanf("%d",&pid);
    ab = find_process(pid);
    if(ab!=NULL){
        free_mem(ab);       // Release the allocation table represented by ab
        dispose(ab);    // Release ab data structure nodes
        return 0;
    }else{
        return -1;
    }
}


// Find if there are partitions that can be non-process allocated

int find_free_mem(int request){
    FBT *tmp = free_block;
    int mem_sum = 0;
    while(tmp){
        if(tmp->size >= request){
            // Can be directly allocated
            return 1;
        }
        mem_sum += tmp->size;
        tmp = tmp->next;
    }
    if(mem_sum >= request){
        // Post-merge allocation
        return 0;
    }else{
        // There is not enough space to allocate
        return -1;
    }


}




// Sort the allocated table from large to small by starting address
void sort_AB(){
    if(allocated_block_head == NULL || allocated_block_head->next == NULL)
        return;
    AB *t1,*t2,*head;
    head = allocated_block_head;
    for(t1 = head->next;t1;t1 = t1->next){
        for(t2 = head;t2 != t1;t2=t2->next){
            if(t2->start_addr > t2->next->start_addr){
                int tmp = t2->start_addr;
                t2->start_addr = t2->next->start_addr;
                t2->next->start_addr = tmp;


                tmp = t2->size;
                t2->size = t2->next->size;
```

```
                    t2->next->size = tmp;
                }
            }
        }
    }


    // Reassign memory addresses to all processes
    void reset_AB(int start){
        /* In a real operating system this is not easy, so memory crunch is not frequently used */
        AB *tmp = allocated_block_head;
        while(tmp != NULL){
            tmp->start_addr = start;
            start += tmp->size;
            tmp = tmp->next;
        }
    }


    void  memory_compact(){
        // Squeeze memory
        FBT *fbttmp = free_block;
        AB *abtmp = allocated_block_head;
        // Detect remaining memory
        int sum = 0;
        while(fbttmp!=NULL){
            sum += fbttmp->size;
            fbttmp = fbttmp->next;
        }


        // Merge blocks into one
        fbttmp = free_block;
        fbttmp->size = sum;
        fbttmp->start_addr = 0;
        fbttmp->next=NULL;

        // Release redundant partitions
        FBT *pr = free_block->next;
        while(pr != NULL){
            fbttmp = pr->next;
            free(pr);
            pr = fbttmp;
        }
        // Reorder the allocated space
        sort_AB();
        reset_AB(sum);


    }


    // Perform memory allocation
    void do_allocate_mem(AB *ab){
        int request = ab->size;
        FBT *tmp = free_block;
        while(tmp != NULL){
            if(tmp->size >= request){
                // allocation
                ab->start_addr = tmp->start_addr;
                int shengyu = tmp->size - request;
                if(shengyu <= min_mem_size){
                    // The surplus is too small to allocate all
                    ab->size = tmp->size;
                    if(tmp == free_block){
                        free_block = free_block->next;
                        free(tmp);
                    }else{
                        FBT *t = free_block;
                        while(t->next != tmp){
                            t = t->next;
                        }
                        t->next = tmp->next;
                        free(tmp);
                    }
                }else{
                    // Cut out the allocated memory
                    tmp->size = shengyu;
                    tmp->start_addr = tmp->start_addr + request;
                }
                return ;
            }
            tmp = tmp->next;
        }
    }


    int allocate_mem(AB *ab){
        /* Allocated memory module */
        FBT *fbt,*pre;
        int request_size=ab->size;
        fbt = pre = free_block;
        /*
        According to the current algorithm, the appropriate free partition is searched in
the linked list of free partition for allocation.
Pay attention to the following situations when allocating:
1. If the free partition can be found and the remaining space after allocation is large enough,
then divide
2. If the free partition can be found and the remaining space after allocation is relatively
small, then allocate it together
3. Find the free partitions that cannot meet the needs, but the sum of the free partitions can
meet the needs.
Then the memory compression technique is adopted to merge the free partitions and then
redistribute them
4. After the successful allocation of memory, the free partition should be kept in order
according to the corresponding algorithm
5. Return 1 if the allocation is successful, otherwise return -1          */
```

```
    // Try to find allocable idle, the results of which are explained in the function

    int f = find_free_mem(request_size);
    if(f == -1){
        // Allocate enough
        printf("Free mem is not enough,Allocate fail!\n");
        return -1;
    }else{
        if(f == 0){
            // Memory crunch is required to allocate
            memory_compact();
        }
        // Perform assigned
        do_allocate_mem(ab);
    }
    // Rearrange the free partitions
    rearrange(ma_algorithm);
    return 1;
}


// Create a new process
int new_process(){
    AB *ab;
    int size;
    int ret;
    ab = (AB*)malloc(sizeof(AB));
    if(!ab) exit(-5);


    ab->next=NULL;
    pid++;
    sprintf(ab->process_name,"PROCESS-%02d",pid);
    ab->pid = pid;
```

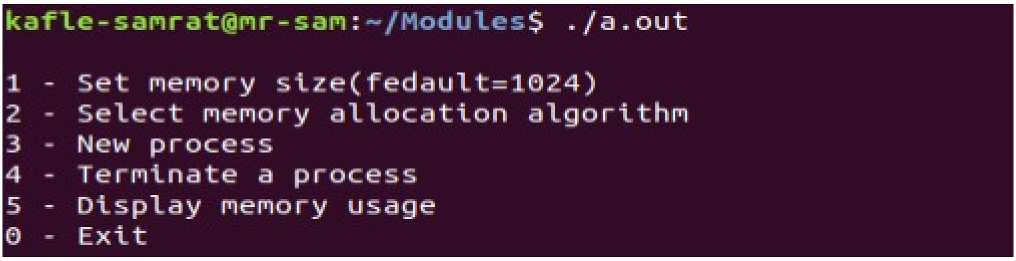
```
        printf("Memory for %s:",ab->process_name);
        scanf("%d",&size);
        if(size>0) ab->size=size;
        ret = allocate_mem(ab);        /* Allocate memory from the free partition, ret==1
indicates successful allocation */

        if((ret == 1) && (allocated_block_head == NULL)){

            allocated_block_head = ab;
            return 1;
        }else if(ret == 1){
            /* Successfully allocate, inserts the description of the allocated block into the
allocated linked list */
            ab->next = allocated_block_head;
            allocated_block_head = ab;
            return 2;
        }else if(ret == -1){
            //分配不成功
            printf("\e[0;31;1m Allocation fail \e[0m\n");
            free(ab);
            return -1;
        }
        return 3;
    }



    void rearrange_FF(){
        /* For the first time, the free zone size is sorted in ascending order according to the
starting address */
        // We use bubble sort here
        if(free_block == NULL || free_block->next == NULL)
            return;
        FBT *t1,*t2,*head;
        head = free_block;
        for(t1 = head->next;t1;t1 = t1->next){
            for(t2 = head;t2 != t1;t2=t2->next){
}
```

```
            if(t2->start_addr > t2->next->start_addr){
                int tmp = t2->start_addr;
                t2->start_addr = t2->next->start_addr;
                t2->next->start_addr = tmp;


                tmp = t2->size;
                t2->size = t2->next->size;
                t2->next->size = tmp;
            }
        }
    }
}


void rearrange_BF(){
    /* The best adaptive algorithm sorts free partitions by size from small to large */
    if(free_block == NULL || free_block->next == NULL)
        return;
    FBT *t1,*t2,*head;
    head = free_block;
    for(t1 = head->next;t1;t1 = t1->next){
        for(t2 = head;t2 != t1;t2=t2->next){
            if(t2->size > t2->next->size){
                int tmp = t2->start_addr;
                t2->start_addr = t2->next->start_addr;
                t2->next->start_addr = tmp;


                tmp = t2->size;
                t2->size = t2->next->size;
                t2->next->size = tmp;
            }
        }
    }
}
```

```
void rearrange_WF(){
    /* The worst fit algorithm sorts free partitions from large to small */
    if(free_block == NULL || free_block->next == NULL)
        return;
    FBT *t1,*t2,*head;
    head = free_block;
    for(t1 = head->next;t1;t1 = t1->next){
        for(t2 = head;t2 != t1;t2=t2->next){
            if(t2->size < t2->next->size){
                int tmp = t2->start_addr;
                t2->start_addr = t2->next->start_addr;
                t2->next->start_addr = tmp;


                tmp = t2->size;
                t2->size = t2->next->size;
                t2->next->size = tmp;
            }
        }
    }
}


/* Collates a list of memory free blocks according to the specified algorithm */
void rearrange(int algorithm){
    switch(algorithm){
        case MA_FF:rearrange_FF();break;
        case MA_BF:rearrange_BF();break;
        case MA_WF:rearrange_WF();break;
    }
}


void set_algorithm(){
    /* Sets the current allocation algorithm */
    int algorithm;
    printf("\t1 - First Fit\n");
    printf("\t2 - Best Fit\n");
    printf("\t3 - Worst Fit\n");
    scanf("%d",&algorithm);
    if(algorithm>=1 && algorithm<=3)
        ma_algorithm = algorithm;


    // Rearrange the list of free areas according to the specified algorithm
    rearrange(ma_algorithm);
}
```

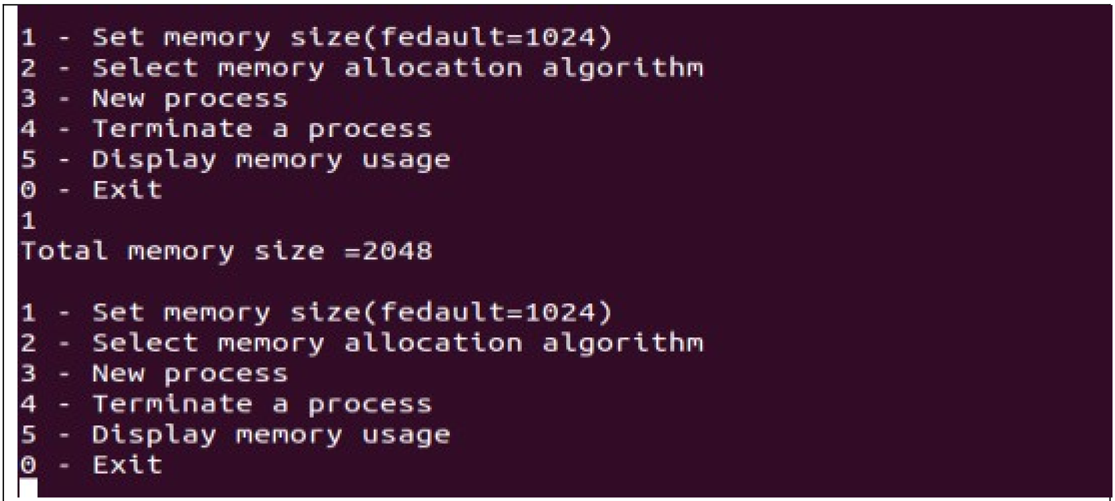
实验结果:

The experimental interface:



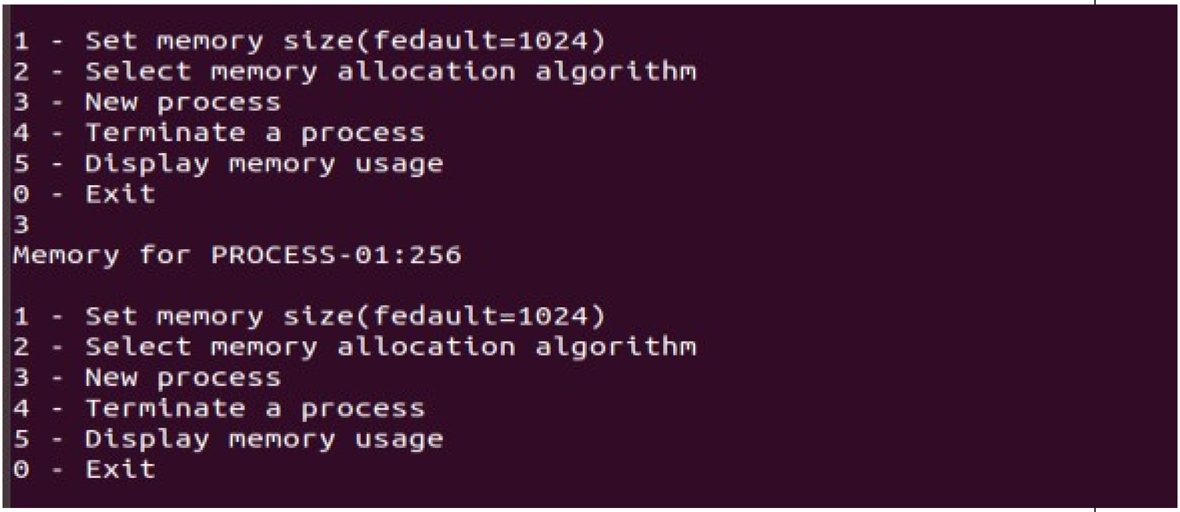
```
kafle-samrat@mr-sam:~/Modules$ ./a.out

1 - Set memory size(fedault=1024)
2 - Select memory allocation algorithm
3 - New process
4 - Terminate a process
5 - Display memory usage
0 - Exit
```

After prompted for input, type 1, and the display is as follows:



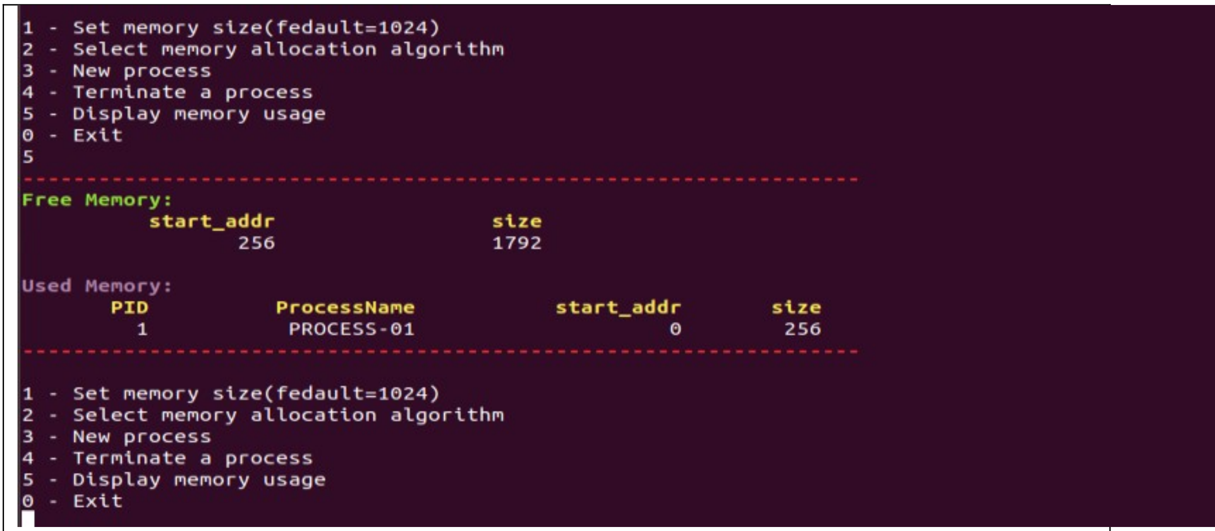
```
1 - Set memory size(fedault=1024)
2 - Select memory allocation algorithm
3 - New process
4 - Terminate a process
5 - Display memory usage
0 - Exit
1
Total memory size =2048

1 - Set memory size(fedault=1024)
2 - Select memory allocation algorithm
3 - New process
4 - Terminate a process
5 - Display memory usage
0 - Exit
```

Then input: 3 and set the memory space to 256.
The display is as follows:



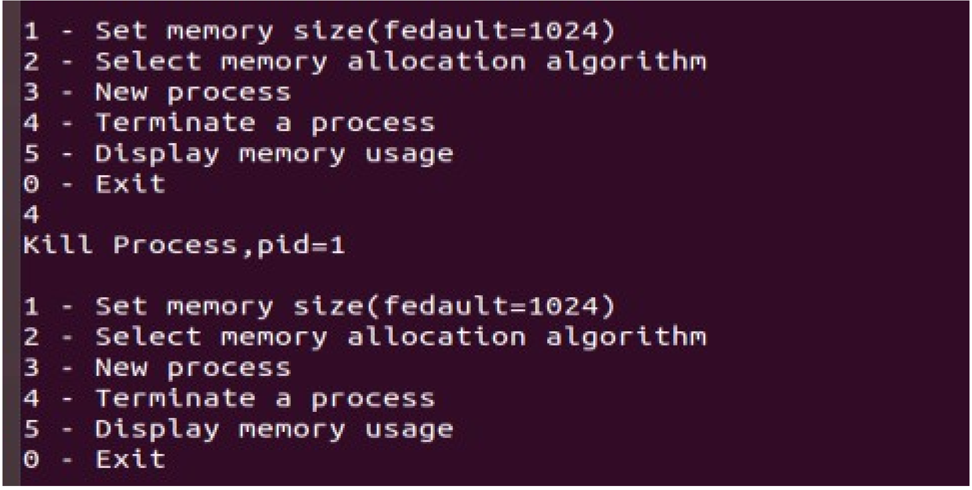
```
1 - Set memory size(fedault=1024)
2 - Select memory allocation algorithm
3 - New process
4 - Terminate a process
5 - Display memory usage
0 - Exit
3
Memory for PROCESS-01:256

1 - Set memory size(fedault=1024)
2 - Select memory allocation algorithm
3 - New process
4 - Terminate a process
5 - Display memory usage
0 - Exit
```

Repeat the previous operation. Input : 5
the display is as follows:



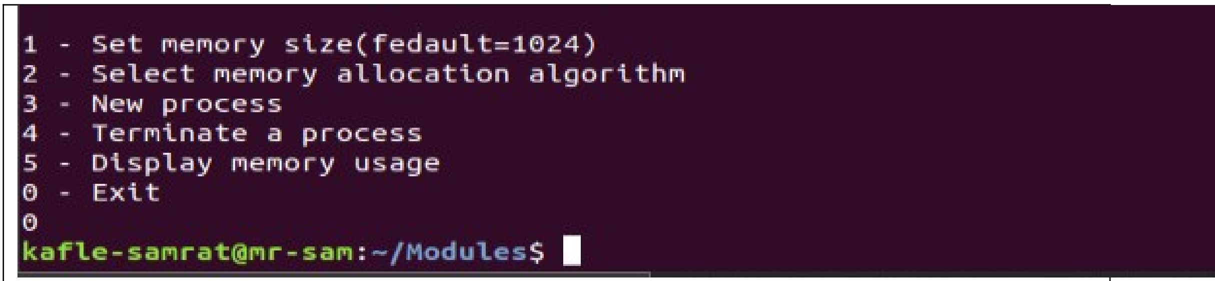
```
1 - Set memory size(fedault=1024)
2 - Select memory allocation algorithm
3 - New process
4 - Terminate a process
5 - Display memory usage
0 - Exit
5
------------------------------------------------------------------
Free Memory:
          start_addr                 size
                 256                 1792

Used Memory:
       PID          ProcessName             start_addr        size
         1           PROCESS-01                      0         256
------------------------------------------------------------------

1 - Set memory size(fedault=1024)
2 - Select memory allocation algorithm
3 - New process
4 - Terminate a process
5 - Display memory usage
0 - Exit
```

After entering : input: 4, kill process 1, as shown below:



```
1 - Set memory size(fedault=1024)
2 - Select memory allocation algorithm
3 - New process
4 - Terminate a process
5 - Display memory usage
0 - Exit
4
Kill Process,pid=1

1 - Set memory size(fedault=1024)
2 - Select memory allocation algorithm
3 - New process
4 - Terminate a process
5 - Display memory usage
0 - Exit
```

Input 0 to Exit the process



```
1 - Set memory size(fedault=1024)
2 - Select memory allocation algorithm
3 - New process
4 - Terminate a process
5 - Display memory usage
0 - Exit
0
kafle-samrat@mr-sam:~/Modules$ 
```

实验心得体会:

通过计算机实验让我进一步了解操作系统对内存分配的知识，也让我认识到C语言的重要性，对记忆分配的方法和思路可以理解，但在具体实现时我感觉有点困难，通过与同学的沟通和咨询相关信息来发现问题，这些都是C语言基础不扎实，而且长期不练习，将来编程练习太重。

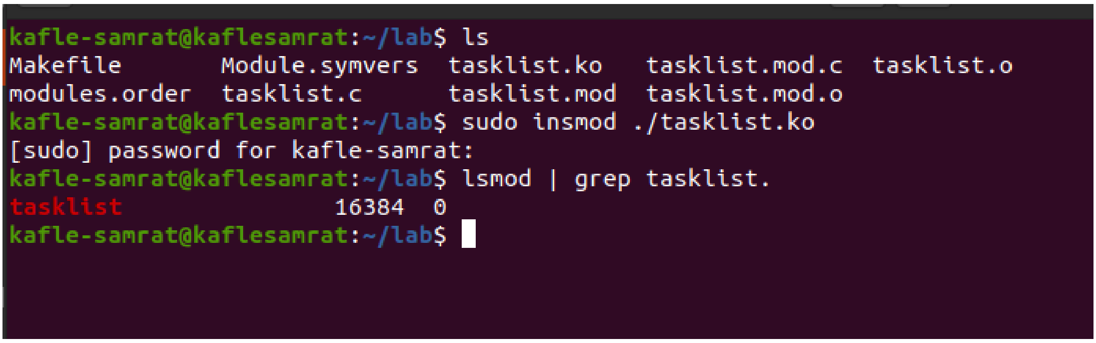
实验报告

组　别		**姓 名**	KAFLE SAMRAT	**同组实验者**	
实验项目名称	linux proc 文件系统查看及进程信息			**实验日期**	月　日
教 师 评 语					
实验成绩：			**指导教师（签名）：** 2020 年　月　日		

一．实验目的

简单了解如何在 Linux 下使用 PROC 文件系统来获取进程信息。通过 PROC 文件系统获取的信息主要是进程使用的虚拟内存，以及 Linux 下的实际内存、信号机制信息等监控工具，可以全面掌握系统的运行情况。

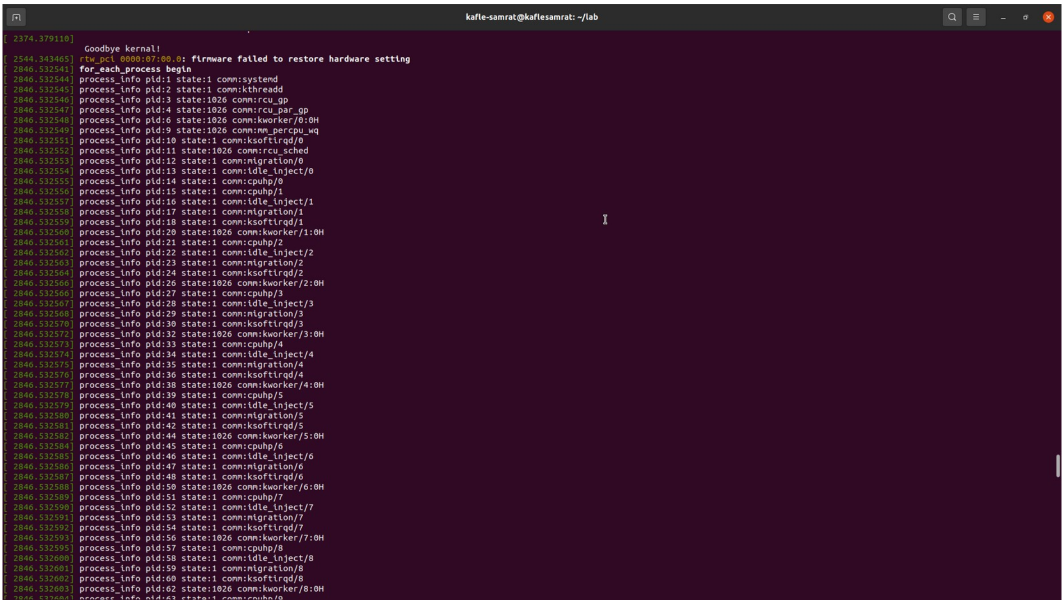
二．实验内容



```
kafle-samrat@kaflesamrat:~/lab$ ls
Makefile       Module.symvers  tasklist.ko   tasklist.mod.c  tasklist.o
modules.order  tasklist.c      tasklist.mod  tasklist.mod.o
kafle-samrat@kaflesamrat:~/lab$ sudo insmod ./tasklist.ko
[sudo] password for kafle-samrat:
kafle-samrat@kaflesamrat:~/lab$ lsmod | grep tasklist.
tasklist               16384  0
kafle-samrat@kaflesamrat:~/lab$ 
```

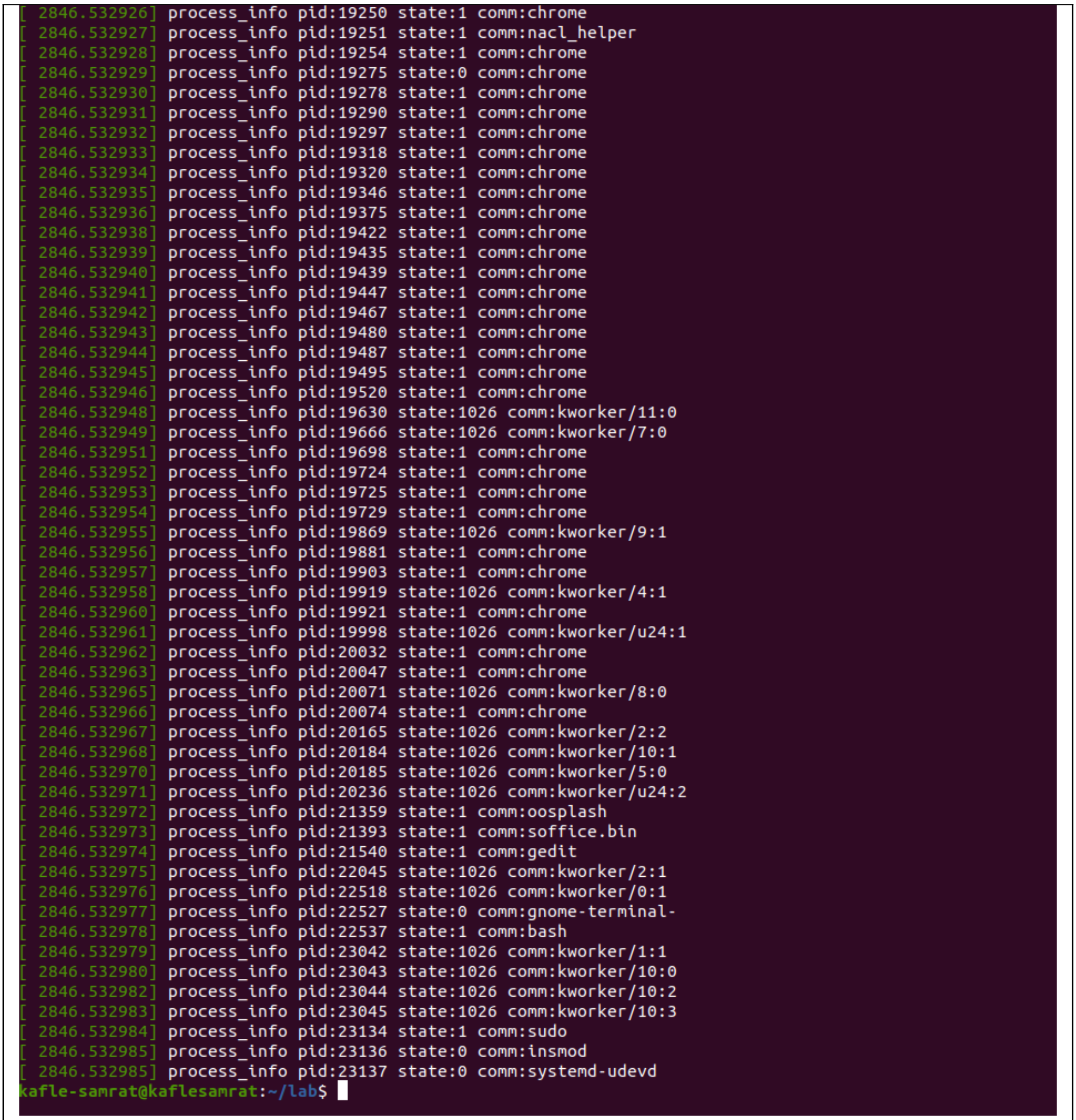
.

sudo dmesg



```
[ 2374.379110]
                Goodbye kernal!
[ 2544.343465] rtw_pci 0000:07:00.0: firmware failed to restore hardware setting
[ 2846.532541] for_each_process begin
[ 2846.532544] process_info pid:1 state:1 comm:systemd
[ 2846.532545] process_info pid:2 state:1 comm:kthreadd
[ 2846.532546] process_info pid:3 state:1026 comm:rcu_gp
[ 2846.532547] process_info pid:4 state:1026 comm:rcu_par_gp
[ 2846.532548] process_info pid:6 state:1026 comm:kworker/0:0H
[ 2846.532549] process_info pid:9 state:1026 comm:mm_percpu_wq
[ 2846.532551] process_info pid:10 state:1 comm:ksoftirqd/0
[ 2846.532552] process_info pid:11 state:1026 comm:rcu_sched
[ 2846.532553] process_info pid:12 state:1 comm:migration/0
[ 2846.532554] process_info pid:13 state:1 comm:idle_inject/0
[ 2846.532555] process_info pid:14 state:1 comm:cpuhp/0
[ 2846.532556] process_info pid:15 state:1 comm:cpuhp/1
[ 2846.532557] process_info pid:16 state:1 comm:idle_inject/1
[ 2846.532558] process_info pid:17 state:1 comm:migration/1
[ 2846.532559] process_info pid:18 state:1 comm:ksoftirqd/1
[ 2846.532560] process_info pid:20 state:1026 comm:kworker/1:0H
[ 2846.532561] process_info pid:21 state:1 comm:cpuhp/2
[ 2846.532562] process_info pid:22 state:1 comm:idle_inject/2
[ 2846.532563] process_info pid:23 state:1 comm:migration/2
[ 2846.532564] process_info pid:24 state:1 comm:ksoftirqd/2
[ 2846.532566] process_info pid:26 state:1026 comm:kworker/2:0H
[ 2846.532566] process_info pid:27 state:1 comm:cpuhp/3
[ 2846.532567] process_info pid:28 state:1 comm:idle_inject/3
[ 2846.532568] process_info pid:29 state:1 comm:migration/3
[ 2846.532570] process_info pid:30 state:1 comm:ksoftirqd/3
[ 2846.532572] process_info pid:32 state:1026 comm:kworker/3:0H
[ 2846.532573] process_info pid:33 state:1 comm:cpuhp/4
[ 2846.532574] process_info pid:34 state:1 comm:idle_inject/4
[ 2846.532575] process_info pid:35 state:1 comm:migration/4
[ 2846.532576] process_info pid:36 state:1 comm:ksoftirqd/4
[ 2846.532577] process_info pid:38 state:1026 comm:kworker/4:0H
[ 2846.532578] process_info pid:39 state:1 comm:cpuhp/5
[ 2846.532579] process_info pid:40 state:1 comm:idle_inject/5
[ 2846.532580] process_info pid:41 state:1 comm:migration/5
[ 2846.532581] process_info pid:42 state:1 comm:ksoftirqd/5
[ 2846.532582] process_info pid:44 state:1026 comm:kworker/5:0H
[ 2846.532584] process_info pid:45 state:1 comm:cpuhp/6
[ 2846.532585] process_info pid:46 state:1 comm:idle_inject/6
[ 2846.532586] process_info pid:47 state:1 comm:migration/6
[ 2846.532587] process_info pid:48 state:1 comm:ksoftirqd/6
[ 2846.532588] process_info pid:50 state:1026 comm:kworker/6:0H
[ 2846.532589] process_info pid:51 state:1 comm:cpuhp/7
[ 2846.532590] process_info pid:52 state:1 comm:idle_inject/7
[ 2846.532591] process_info pid:53 state:1 comm:migration/7
[ 2846.532592] process_info pid:54 state:1 comm:ksoftirqd/7
[ 2846.532593] process_info pid:56 state:1026 comm:kworker/7:0H
[ 2846.532595] process_info pid:57 state:1 comm:cpuhp/8
[ 2846.532600] process_info pid:58 state:1 comm:idle_inject/8
[ 2846.532601] process_info pid:59 state:1 comm:migration/8
[ 2846.532602] process_info pid:60 state:1 comm:ksoftirqd/8
[ 2846.532603] process_info pid:62 state:1026 comm:kworker/8:0H
[ 2846.532604] process_info pid:63 state:1 comm:cpuhp/9
```

.



```
[ 2846.532926] process_info pid:19250 state:1 comm:chrome
[ 2846.532927] process_info pid:19251 state:1 comm:nacl_helper
[ 2846.532928] process_info pid:19254 state:1 comm:chrome
[ 2846.532929] process_info pid:19275 state:0 comm:chrome
[ 2846.532930] process_info pid:19278 state:1 comm:chrome
[ 2846.532931] process_info pid:19290 state:1 comm:chrome
[ 2846.532932] process_info pid:19297 state:1 comm:chrome
[ 2846.532933] process_info pid:19318 state:1 comm:chrome
[ 2846.532934] process_info pid:19320 state:1 comm:chrome
[ 2846.532935] process_info pid:19346 state:1 comm:chrome
[ 2846.532936] process_info pid:19375 state:1 comm:chrome
[ 2846.532938] process_info pid:19422 state:1 comm:chrome
[ 2846.532939] process_info pid:19435 state:1 comm:chrome
[ 2846.532940] process_info pid:19439 state:1 comm:chrome
[ 2846.532941] process_info pid:19447 state:1 comm:chrome
[ 2846.532942] process_info pid:19467 state:1 comm:chrome
[ 2846.532943] process_info pid:19480 state:1 comm:chrome
[ 2846.532944] process_info pid:19487 state:1 comm:chrome
[ 2846.532945] process_info pid:19495 state:1 comm:chrome
[ 2846.532946] process_info pid:19520 state:1 comm:chrome
[ 2846.532948] process_info pid:19630 state:1026 comm:kworker/11:0
[ 2846.532949] process_info pid:19666 state:1026 comm:kworker/7:0
[ 2846.532951] process_info pid:19698 state:1 comm:chrome
[ 2846.532952] process_info pid:19724 state:1 comm:chrome
[ 2846.532953] process_info pid:19725 state:1 comm:chrome
[ 2846.532954] process_info pid:19729 state:1 comm:chrome
[ 2846.532955] process_info pid:19869 state:1026 comm:kworker/9:1
[ 2846.532956] process_info pid:19881 state:1 comm:chrome
[ 2846.532957] process_info pid:19903 state:1 comm:chrome
[ 2846.532958] process_info pid:19919 state:1026 comm:kworker/4:1
[ 2846.532960] process_info pid:19921 state:1 comm:chrome
[ 2846.532961] process_info pid:19998 state:1026 comm:kworker/u24:1
[ 2846.532962] process_info pid:20032 state:1 comm:chrome
[ 2846.532963] process_info pid:20047 state:1 comm:chrome
[ 2846.532965] process_info pid:20071 state:1026 comm:kworker/8:0
[ 2846.532966] process_info pid:20074 state:1 comm:chrome
[ 2846.532967] process_info pid:20165 state:1026 comm:kworker/2:2
[ 2846.532968] process_info pid:20184 state:1026 comm:kworker/10:1
[ 2846.532970] process_info pid:20185 state:1026 comm:kworker/5:0
[ 2846.532971] process_info pid:20236 state:1026 comm:kworker/u24:2
[ 2846.532972] process_info pid:21359 state:1 comm:oosplash
[ 2846.532973] process_info pid:21393 state:1 comm:soffice.bin
[ 2846.532974] process_info pid:21540 state:1 comm:gedit
[ 2846.532975] process_info pid:22045 state:1026 comm:kworker/2:1
[ 2846.532976] process_info pid:22518 state:1026 comm:kworker/0:1
[ 2846.532977] process_info pid:22527 state:0 comm:gnome-terminal-
[ 2846.532978] process_info pid:22537 state:1 comm:bash
[ 2846.532979] process_info pid:23042 state:1026 comm:kworker/1:1
[ 2846.532980] process_info pid:23043 state:1026 comm:kworker/10:0
[ 2846.532982] process_info pid:23044 state:1026 comm:kworker/10:2
[ 2846.532983] process_info pid:23045 state:1026 comm:kworker/10:3
[ 2846.532984] process_info pid:23134 state:1 comm:sudo
[ 2846.532985] process_info pid:23136 state:0 comm:insmod
[ 2846.532985] process_info pid:23137 state:0 comm:systemd-udevd
kafle-samrat@kaflesamrat:~/lab$
```

.

code : tasklist.c

```
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/proc_fs.h>
#include <linux/sched/signal.h>
#include <linux/init.h>
static int __init hello_init(void)
{
    struct task_struct *pp;
    printk("for_each_process begin\n");
    for_each_process(pp)
    {
        printk(KERN_INFO "process_info pid:%i state:
%lu comm:%s \n",pp->pid,pp->state,pp->comm);
        }
    return 0;
}
static void __exit hello_exit(void)
{
    printk("for_each_process end!\n");
}
module_init(hello_init);
module_exit(hello_exit);
MODULE_LICENSE("GPL");
```

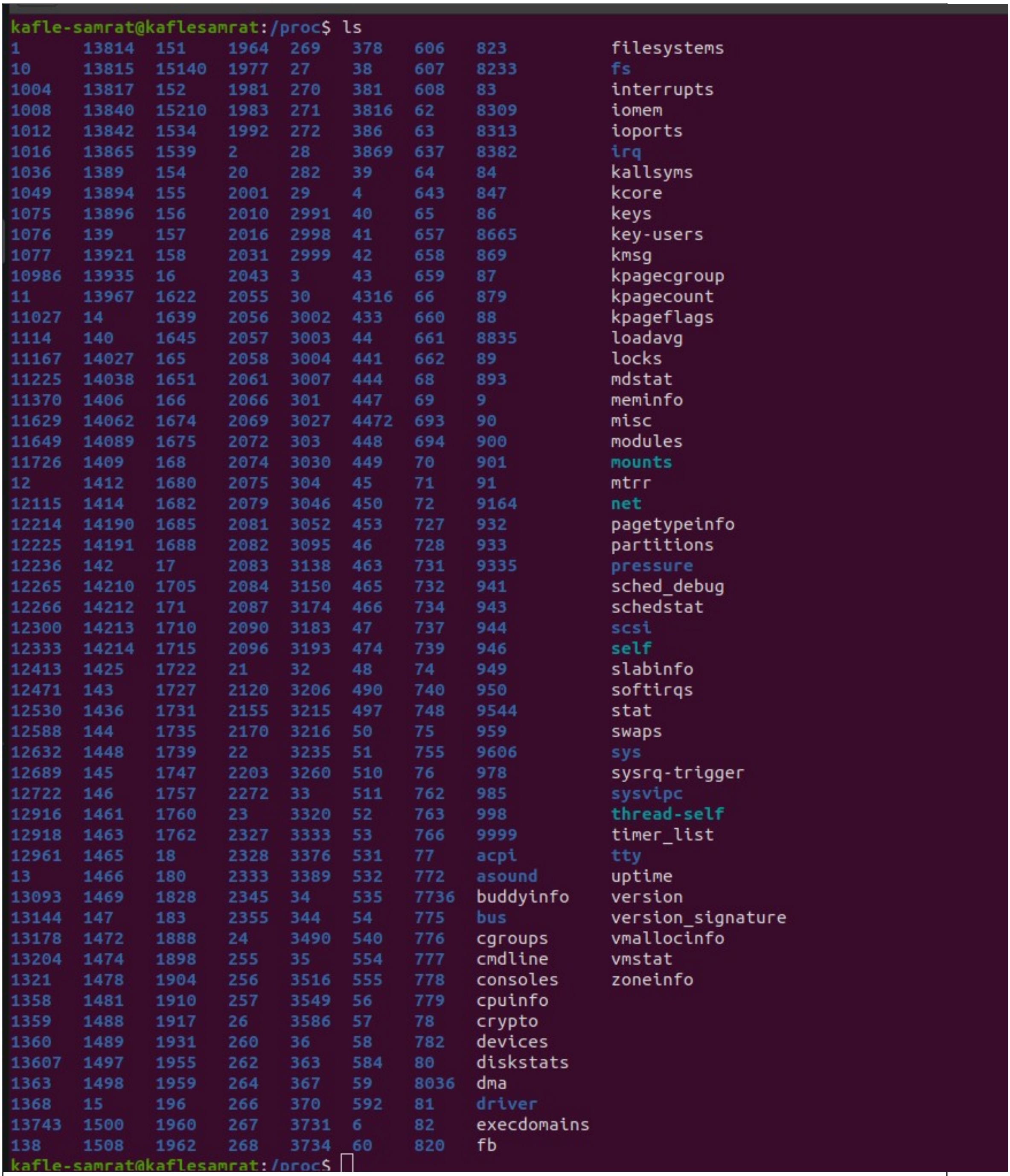
mod.c

```
#include <linux/build-salt.h>
#include <linux/module.h>
#include <linux/vermagic.h>
#include <linux/compiler.h>

BUILD_SALT;

MODULE_INFO(vermagic, VERMAGIC_STRING);
MODULE_INFO(name, KBUILD_MODNAME);

__visible struct module __this_module
__section(.gnu.linkonce.this_module) = {
   .name = KBUILD_MODNAME,
   .init = init_module,
#ifdef CONFIG_MODULE_UNLOAD
   .exit = cleanup_module,
#endif
   .arch = MODULE_ARCH_INIT,
};

#ifdef CONFIG_RETPOLINE
MODULE_INFO(retpoline, "Y");
#endif

MODULE_INFO(depends, "");


MODULE_INFO(srcversion, "F79FE6886324425D89377F9");



make file :

obj-m := tasklist.o
```

```
KVERSION = $(shell uname -r)

KERNELDR :=/lib/modules/$(KVERSION)/build

PWD := $(shell pwd)

modules:

   $(MAKE) -C $(KERNELDR) M=$(PWD) modules

moduels_install:

   $(MAKE) -C $(KERNELDR) M=$(PWD) modules_install

clean:

   rm -rf *.o *~ core .depend .*.cmd *.ko

*.mod.c .tmp_versions
```

```
ls /proc
```

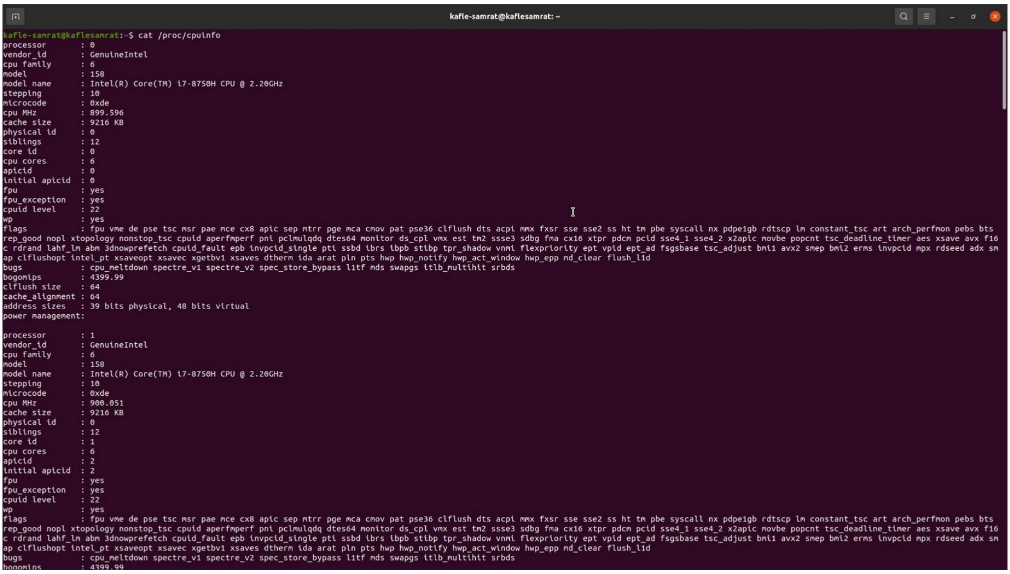
View the contents of the /proc directory



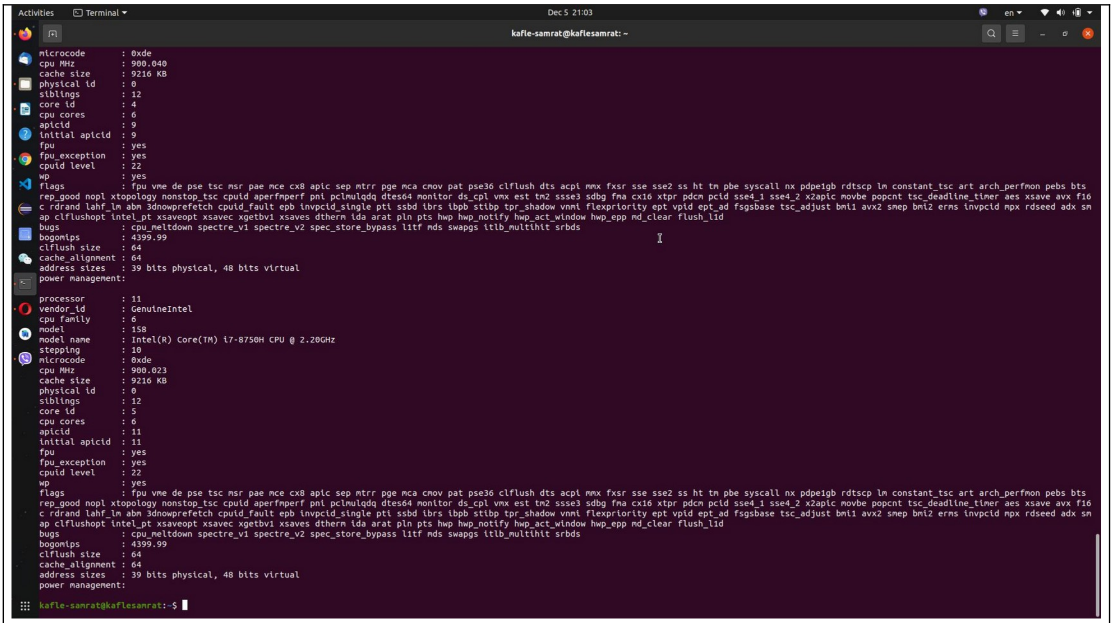
```
kafle-samrat@kaflesamrat:/proc$ ls
1      13814  151    1964  269   378   606   823          filesystems
10     13815  15140  1977  27    38    607   8233         fs
1004   13817  152    1981  270   381   608   83           interrupts
1008   13840  15210  1983  271   3816  62    8309         iomem
1012   13842  1534   1992  272   386   63    8313         ioports
1016   13865  1539   2     28    3869  637   8382         irq
1036   1389   154    20    282   39    64    84           kallsyms
1049   13894  155    2001  29    4     643   847          kcore
1075   13896  156    2010  2991  40    65    86           keys
1076   139    157    2016  2998  41    657   8665         key-users
1077   13921  158    2031  2999  42    658   869          kmsg
10986  13935  16     2043  3     43    659   87           kpagecgroup
11     13967  1622   2055  30    4316  66    879          kpagecount
11027  14     1639   2056  3002  433   660   88           kpageflags
1114   140    1645   2057  3003  44    661   8835         loadavg
11167  14027  165    2058  3004  441   662   89           locks
11225  14038  1651   2061  3007  444   68    893          mdstat
11370  1406   166    2066  301   447   69    9            meminfo
11629  14062  1674   2069  3027  4472  693   90           misc
11649  14089  1675   2072  303   448   694   900          modules
11726  1409   168    2074  3030  449   70    901          mounts
12     1412   1680   2075  304   45    71    91           mtrr
12115  1414   1682   2079  3046  450   72    9164         net
12214  14190  1685   2081  3052  453   727   932          pagetypeinfo
12225  14191  1688   2082  3095  46    728   933          partitions
12236  142    17     2083  3138  463   731   9335         pressure
12265  14210  1705   2084  3150  465   732   941          sched_debug
12266  14212  171    2087  3174  466   734   943          schedstat
12300  14213  1710   2090  3183  47    737   944          scsi
12333  14214  1715   2096  3193  474   739   946          self
12413  1425   1722   21    32    48    74    949          slabinfo
12471  143    1727   2120  3206  490   740   950          softirqs
12530  1436   1731   2155  3215  497   748   9544         stat
12588  144    1735   2170  3216  50    75    959          swaps
12632  1448   1739   22    3235  51    755   9606         sys
12689  145    1747   2203  3260  510   76    978          sysrq-trigger
12722  146    1757   2272  33    511   762   985          sysvipc
12916  1461   1760   23    3320  52    763   998          thread-self
12918  1463   1762   2327  3333  53    766   9999         timer_list
12961  1465   18     2328  3376  531   77    acpi         tty
13     1466   180    2333  3389  532   772   asound       uptime
13093  1469   1828   2345  34    535   7736  buddyinfo    version
13144  147    183    2355  344   54    775   bus          version_signature
13178  1472   1888   24    3490  540   776   cgroups      vmallocinfo
13204  1474   1898   255   35    554   777   cmdline      vmstat
1321   1478   1904   256   3516  555   778   consoles     zoneinfo
1358   1481   1910   257   3549  56    779   cpuinfo
1359   1488   1917   26    3586  57    78    crypto
1360   1489   1931   260   36    58    782   devices
13607  1497   1955   262   363   584   80    diskstats
1363   1498   1959   264   367   59    8036  dma
1368   15     196    266   370   592   81    driver
13743  1500   1960   267   3731  6     82    execdomains
138    1508   1962   268   3734  60    820   fb
kafle-samrat@kaflesamrat:/proc$ 
```

.

cat /proc/cpuinfo   check cpu info

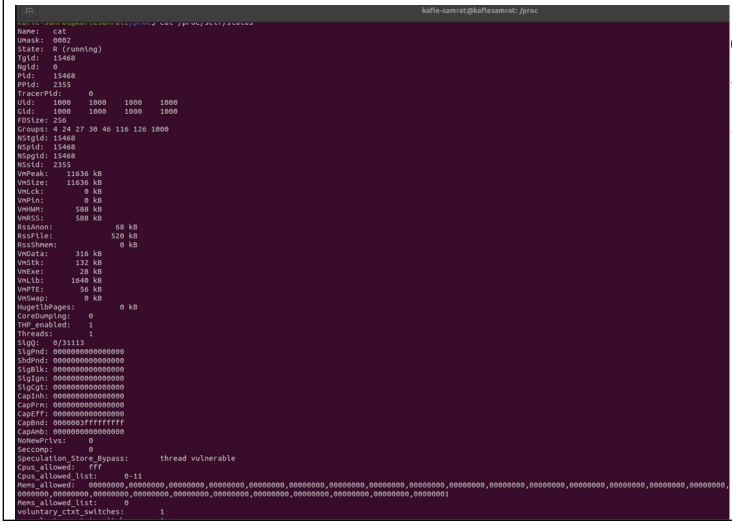


```
kafle-samrat@kaflesamrat:~$ cat /proc/cpuinfo
processor       : 0
vendor_id       : GenuineIntel
cpu family      : 6
model           : 158
model name      : Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz
stepping        : 10
microcode       : 0xde
cpu MHz         : 899.596
cache size      : 9216 KB
physical id     : 0
siblings        : 12
core id         : 0
cpu cores       : 6
apicid          : 0
initial apicid  : 0
fpu             : yes
fpu_exception   : yes
cpuid level     : 22
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts
rep_good nopl xtopology nonstop_tsc cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16
c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb invpcid_single pti ssbd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 avx2 smep bmi2 erms invpcid mpx rdseed adx sm
ap clflushopt intel_pt xsaveopt xsavec xgetbv1 xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp md_clear flush_l1d
bugs            : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs itlb_multihit srbds
bogomips        : 4399.99
clflush size    : 64
cache_alignment : 64
address sizes   : 39 bits physical, 48 bits virtual
power management:

processor       : 1
vendor_id       : GenuineIntel
cpu family      : 6
model           : 158
model name      : Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz
stepping        : 10
microcode       : 0xde
cpu MHz         : 900.051
cache size      : 9216 KB
physical id     : 0
siblings        : 12
core id         : 1
cpu cores       : 6
apicid          : 2
initial apicid  : 2
fpu             : yes
fpu_exception   : yes
cpuid level     : 22
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts
rep_good nopl xtopology nonstop_tsc cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16
c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb invpcid_single pti ssbd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 avx2 smep bmi2 erms invpcid mpx rdseed adx sm
ap clflushopt intel_pt xsaveopt xsavec xgetbv1 xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp md_clear flush_l1d
bugs            : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs itlb_multihit srbds
bogomips        : 4399.99
```



```
microcode       : 0xde
cpu MHz         : 900.040
cache size      : 9216 KB
physical id     : 0
siblings        : 12
core id         : 4
cpu cores       : 6
apicid          : 9
initial apicid  : 9
fpu             : yes
fpu_exception   : yes
cpuid level     : 22
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts
rep_good nopl xtopology nonstop_tsc cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16
c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb invpcid_single pti ssbd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 avx2 smep bmi2 erms invpcid mpx rdseed adx sm
ap clflushopt intel_pt xsaveopt xsavec xgetbv1 xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp md_clear flush_l1d
bugs            : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs itlb_multihit srbds
bogomips        : 4399.99
clflush size    : 64
cache_alignment : 64
address sizes   : 39 bits physical, 48 bits virtual
power management:

processor       : 11
vendor_id       : GenuineIntel
cpu family      : 6
model           : 158
model name      : Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz
stepping        : 10
microcode       : 0xde
cpu MHz         : 900.023
cache size      : 9216 KB
physical id     : 0
siblings        : 12
core id         : 5
cpu cores       : 6
apicid          : 11
initial apicid  : 11
fpu             : yes
fpu_exception   : yes
cpuid level     : 22
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc art arch_perfmon pebs bts
rep_good nopl xtopology nonstop_tsc cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16
c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb invpcid_single pti ssbd ibrs ibpb stibp tpr_shadow vnmi flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 avx2 smep bmi2 erms invpcid mpx rdseed adx sm
ap clflushopt intel_pt xsaveopt xsavec xgetbv1 xsaves dtherm ida arat pln pts hwp hwp_notify hwp_act_window hwp_epp md_clear flush_l1d
bugs            : cpu_meltdown spectre_v1 spectre_v2 spec_store_bypass l1tf mds swapgs itlb_multihit srbds
bogomips        : 4399.99
clflush size    : 64
cache_alignment : 64
address sizes   : 39 bits physical, 48 bits virtual
power management:

kafle-samrat@kaflesamrat:~$ 
```

-

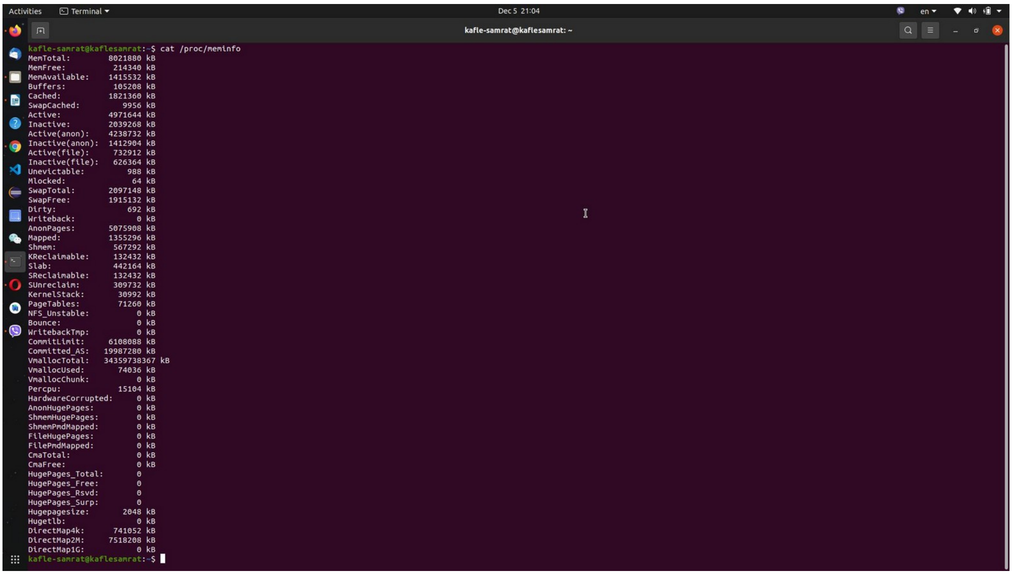


```
kafle-samrat@kaflesamrat:/proc$ cat /proc/self/status
Name:   cat
Umask:  0002
State:  R (running)
Tgid:   15468
Ngid:   0
Pid:    15468
PPid:   2355
TracerPid:      0
Uid:    1000    1000    1000    1000
Gid:    1000    1000    1000    1000
FDSize: 256
Groups: 4 24 27 30 46 116 126 1000
NStgid: 15468
NSpid:  15468
NSpgid: 15468
NSsid:  2355
VmPeak:    11636 kB
VmSize:    11636 kB
VmLck:         0 kB
VmPin:         0 kB
VmHWM:       588 kB
VmRSS:       588 kB
RssAnon:              68 kB
RssFile:             520 kB
RssShmem:              0 kB
VmData:      316 kB
VmStk:       132 kB
VmExe:        28 kB
VmLib:      1640 kB
VmPTE:        56 kB
VmSwap:        0 kB
HugetlbPages:          0 kB
CoreDumping:    0
THP_enabled:    1
Threads:        1
SigQ:   0/31113
SigPnd: 0000000000000000
ShdPnd: 0000000000000000
SigBlk: 0000000000000000
SigIgn: 0000000000000000
SigCgt: 0000000000000000
CapInh: 0000000000000000
CapPrm: 0000000000000000
CapEff: 0000000000000000
CapBnd: 0000003fffffffff
CapAmb: 0000000000000000
NoNewPrivs:     0
Seccomp:        0
Speculation_Store_Bypass:       thread vulnerable
Cpus_allowed:   fff
Cpus_allowed_list:      0-11
Mems_allowed:   00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000,
0000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000000,00000001
Mems_allowed_list:      0
voluntary_ctxt_switches:        1
```

Cat /proc/self/status view the current status of the system

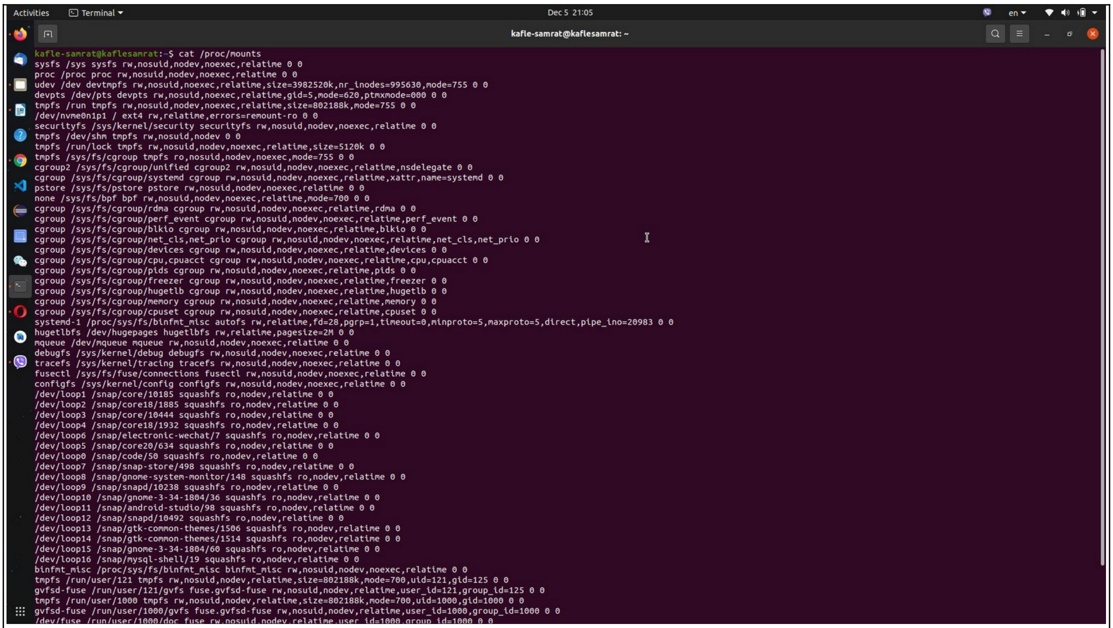
-

/proc/meminfo and converts the number of bytes given to kilobytes



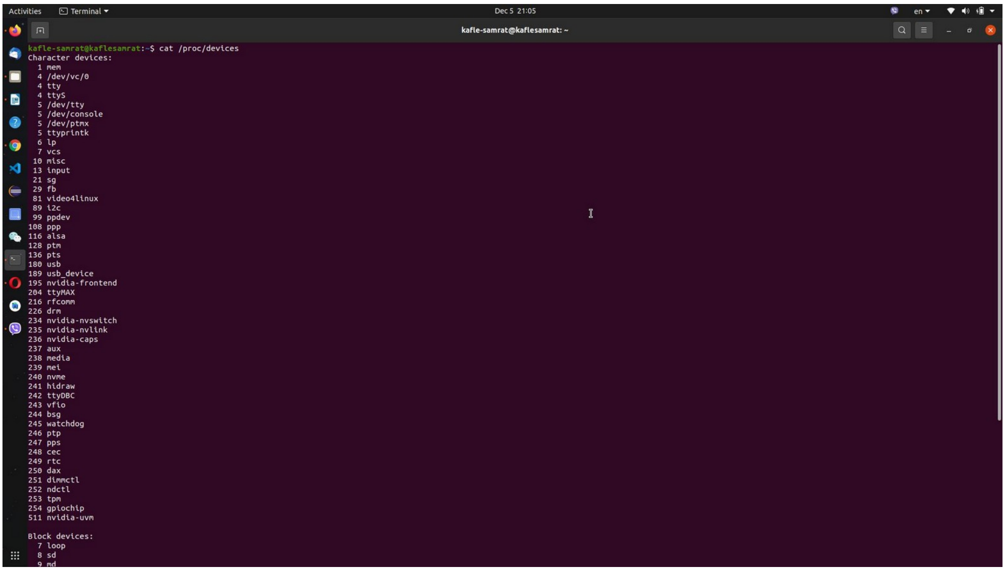
```
kafle-samrat@kaflesamrat:~$ cat /proc/meminfo
MemTotal:        8021880 kB
MemFree:          214340 kB
MemAvailable:    1415532 kB
Buffers:          105208 kB
Cached:          1821360 kB
SwapCached:         9956 kB
Active:          4971644 kB
Inactive:        2039268 kB
Active(anon):    4238732 kB
Inactive(anon):  1412904 kB
Active(file):     732912 kB
Inactive(file):   626364 kB
Unevictable:         988 kB
Mlocked:              64 kB
SwapTotal:       2097148 kB
SwapFree:        1915132 kB
Dirty:               692 kB
Writeback:             0 kB
AnonPages:       5075908 kB
Mapped:          1355296 kB
Shmem:            567292 kB
KReclaimable:     132432 kB
Slab:             442164 kB
SReclaimable:     132432 kB
SUnreclaim:       309732 kB
KernelStack:       30992 kB
PageTables:        71260 kB
NFS_Unstable:          0 kB
Bounce:                0 kB
WritebackTmp:          0 kB
CommitLimit:     6108088 kB
Committed_AS:   19987280 kB
VmallocTotal:   34359738367 kB
VmallocUsed:       74036 kB
VmallocChunk:          0 kB
Percpu:            15104 kB
HardwareCorrupted:     0 kB
AnonHugePages:         0 kB
ShmemHugePages:        0 kB
ShmemPmdMapped:        0 kB
FileHugePages:         0 kB
FilePmdMapped:         0 kB
CmaTotal:              0 kB
CmaFree:               0 kB
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       2048 kB
Hugetlb:               0 kB
DirectMap4k:      741052 kB
DirectMap2M:     7518208 kB
DirectMap1G:           0 kB
kafle-samrat@kaflesamrat:~$ 
```

cat *proc/*mounts   Table of the file system that has been loaded



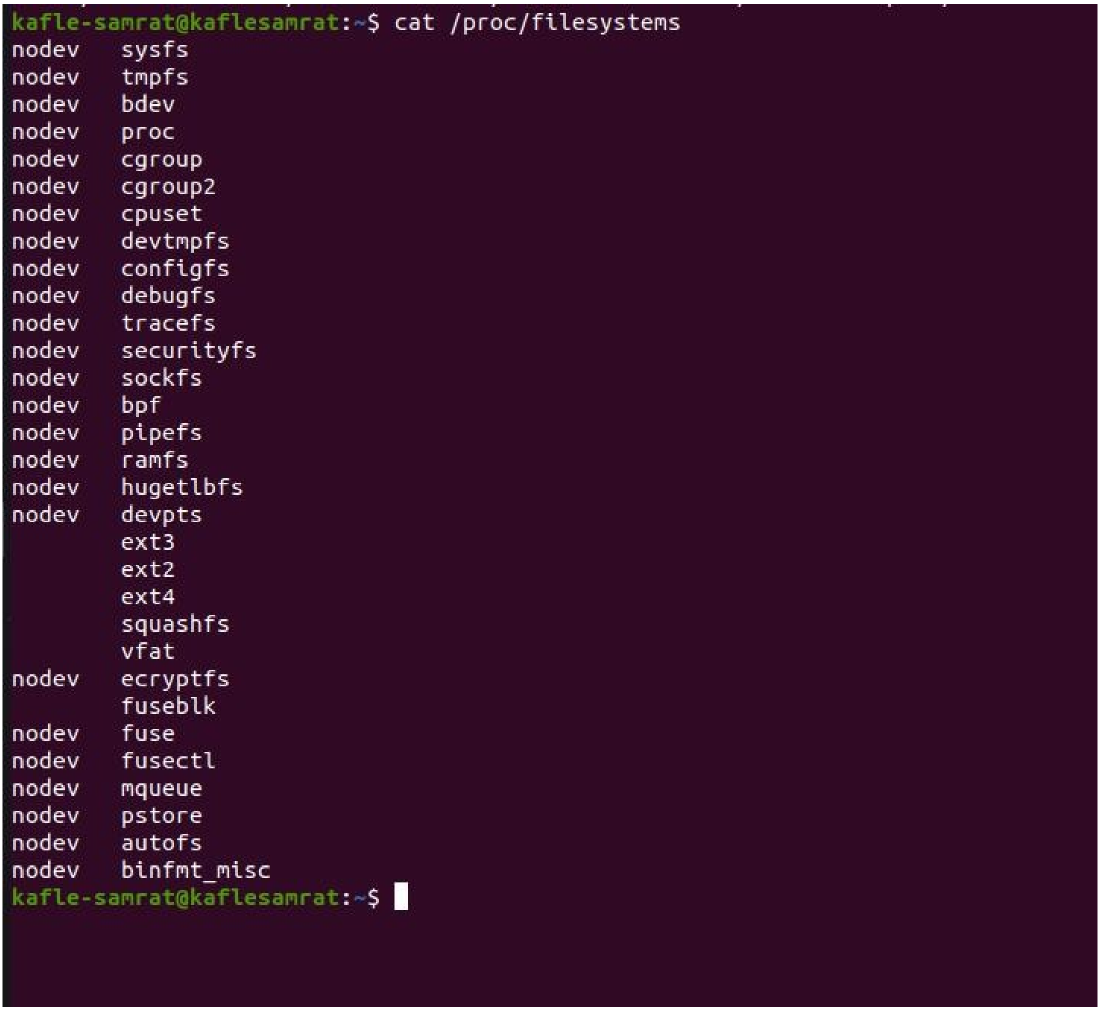
```
kafle-samrat@kaflesamrat:~$ cat /proc/mounts
sysfs /sys sysfs rw,nosuid,nodev,noexec,relatime 0 0
proc /proc proc rw,nosuid,nodev,noexec,relatime 0 0
udev /dev devtmpfs rw,nosuid,noexec,relatime,size=3982520k,nr_inodes=995630,mode=755 0 0
devpts /dev/pts devpts rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000 0 0
tmpfs /run tmpfs rw,nosuid,nodev,noexec,relatime,size=802188k,mode=755 0 0
/dev/nvme0n1p1 / ext4 rw,relatime,errors=remount-ro 0 0
securityfs /sys/kernel/security securityfs rw,nosuid,nodev,noexec,relatime 0 0
tmpfs /dev/shm tmpfs rw,nosuid,nodev 0 0
tmpfs /run/lock tmpfs rw,nosuid,nodev,noexec,relatime,size=5120k 0 0
tmpfs /sys/fs/cgroup tmpfs ro,nosuid,nodev,noexec,mode=755 0 0
cgroup2 /sys/fs/cgroup/unified cgroup2 rw,nosuid,nodev,noexec,relatime,nsdelegate 0 0
cgroup /sys/fs/cgroup/systemd cgroup rw,nosuid,nodev,noexec,relatime,xattr,name=systemd 0 0
pstore /sys/fs/pstore pstore rw,nosuid,nodev,noexec,relatime 0 0
none /sys/fs/bpf bpf rw,nosuid,nodev,noexec,relatime,mode=700 0 0
cgroup /sys/fs/cgroup/rdma cgroup rw,nosuid,nodev,noexec,relatime,rdma 0 0
cgroup /sys/fs/cgroup/perf_event cgroup rw,nosuid,nodev,noexec,relatime,perf_event 0 0
cgroup /sys/fs/cgroup/blkio cgroup rw,nosuid,nodev,noexec,relatime,blkio 0 0
cgroup /sys/fs/cgroup/net_cls,net_prio cgroup rw,nosuid,nodev,noexec,relatime,net_cls,net_prio 0 0
cgroup /sys/fs/cgroup/devices cgroup rw,nosuid,nodev,noexec,relatime,devices 0 0
cgroup /sys/fs/cgroup/cpu,cpuacct cgroup rw,nosuid,nodev,noexec,relatime,cpu,cpuacct 0 0
cgroup /sys/fs/cgroup/pids cgroup rw,nosuid,nodev,noexec,relatime,pids 0 0
cgroup /sys/fs/cgroup/freezer cgroup rw,nosuid,nodev,noexec,relatime,freezer 0 0
cgroup /sys/fs/cgroup/hugetlb cgroup rw,nosuid,nodev,noexec,relatime,hugetlb 0 0
cgroup /sys/fs/cgroup/memory cgroup rw,nosuid,nodev,noexec,relatime,memory 0 0
cgroup /sys/fs/cgroup/cpuset cgroup rw,nosuid,nodev,noexec,relatime,cpuset 0 0
systemd-1 /proc/sys/fs/binfmt_misc autofs rw,relatime,fd=28,pgrp=1,timeout=0,minproto=5,maxproto=5,direct,pipe_ino=20983 0 0
hugetlbfs /dev/hugepages hugetlbfs rw,relatime,pagesize=2M 0 0
mqueue /dev/mqueue mqueue rw,nosuid,nodev,noexec,relatime 0 0
debugfs /sys/kernel/debug debugfs rw,nosuid,nodev,noexec,relatime 0 0
tracefs /sys/kernel/tracing tracefs rw,nosuid,nodev,noexec,relatime 0 0
fusectl /sys/fs/fuse/connections fusectl rw,nosuid,nodev,noexec,relatime 0 0
configfs /sys/kernel/config configfs rw,nosuid,nodev,noexec,relatime 0 0
/dev/loop1 /snap/core/10185 squashfs ro,nodev,relatime 0 0
/dev/loop2 /snap/core18/1885 squashfs ro,nodev,relatime 0 0
/dev/loop3 /snap/core/10444 squashfs ro,nodev,relatime 0 0
/dev/loop4 /snap/core18/1932 squashfs ro,nodev,relatime 0 0
/dev/loop6 /snap/electronic-wechat/7 squashfs ro,nodev,relatime 0 0
/dev/loop5 /snap/core20/634 squashfs ro,nodev,relatime 0 0
/dev/loop0 /snap/code/50 squashfs ro,nodev,relatime 0 0
/dev/loop7 /snap/snap-store/498 squashfs ro,nodev,relatime 0 0
/dev/loop8 /snap/gnome-system-monitor/148 squashfs ro,nodev,relatime 0 0
/dev/loop9 /snap/snapd/10238 squashfs ro,nodev,relatime 0 0
/dev/loop10 /snap/gnome-3-34-1804/36 squashfs ro,nodev,relatime 0 0
/dev/loop11 /snap/android-studio/98 squashfs ro,nodev,relatime 0 0
/dev/loop12 /snap/snapd/10492 squashfs ro,nodev,relatime 0 0
/dev/loop13 /snap/gtk-common-themes/1506 squashfs ro,nodev,relatime 0 0
/dev/loop14 /snap/gtk-common-themes/1514 squashfs ro,nodev,relatime 0 0
/dev/loop15 /snap/gnome-3-34-1804/60 squashfs ro,nodev,relatime 0 0
/dev/loop16 /snap/mysql-shell/19 squashfs ro,nodev,relatime 0 0
binfmt_misc /proc/sys/fs/binfmt_misc binfmt_misc rw,nosuid,nodev,noexec,relatime 0 0
tmpfs /run/user/121 tmpfs rw,nosuid,nodev,relatime,size=802188k,mode=700,uid=121,gid=125 0 0
gvfsd-fuse /run/user/121/gvfs fuse.gvfsd-fuse rw,nosuid,nodev,relatime,user_id=121,group_id=125 0 0
tmpfs /run/user/1000 tmpfs rw,nosuid,nodev,relatime,size=802188k,mode=700,uid=1000,gid=1000 0 0
gvfsd-fuse /run/user/1000/gvfs fuse.gvfsd-fuse rw,nosuid,nodev,relatime,user_id=1000,group_id=1000 0 0
/dev/fuse /run/user/1000/doc fuse rw,nosuid,nodev,relatime,user_id=1000,group_id=1000 0 0
```

cat *proc*devices   uses devices info



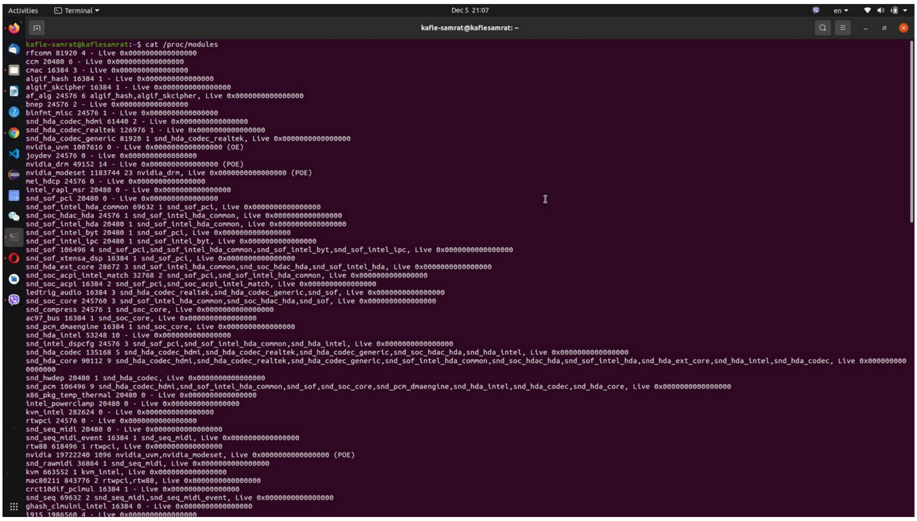
```
kafle-samrat@kaflesamrat:~$ cat /proc/devices
Character devices:
  1 mem
  4 /dev/vc/0
  4 tty
  4 ttyS
  5 /dev/tty
  5 /dev/console
  5 /dev/ptmx
  5 ttyprintk
  6 lp
  7 vcs
 10 misc
 13 input
 21 sg
 29 fb
 81 video4linux
 89 i2c
 99 ppdev
108 ppp
116 alsa
128 ptm
136 pts
180 usb
189 usb_device
195 nvidia-frontend
204 ttyMAX
216 rfcomm
226 drm
234 nvidia-nvswitch
235 nvidia-nvlink
236 nvidia-caps
237 aux
238 media
239 mei
240 nvme
241 hidraw
242 ttyDBC
243 vfio
244 bsg
245 watchdog
246 ptp
247 pps
248 cec
249 rtc
250 dax
251 dimmctl
252 ndctl
253 tpm
254 gpiochip
511 nvidia-uvm

Block devices:
  7 loop
  8 sd
  9 md
```

cat *proc/*filesystems check file system

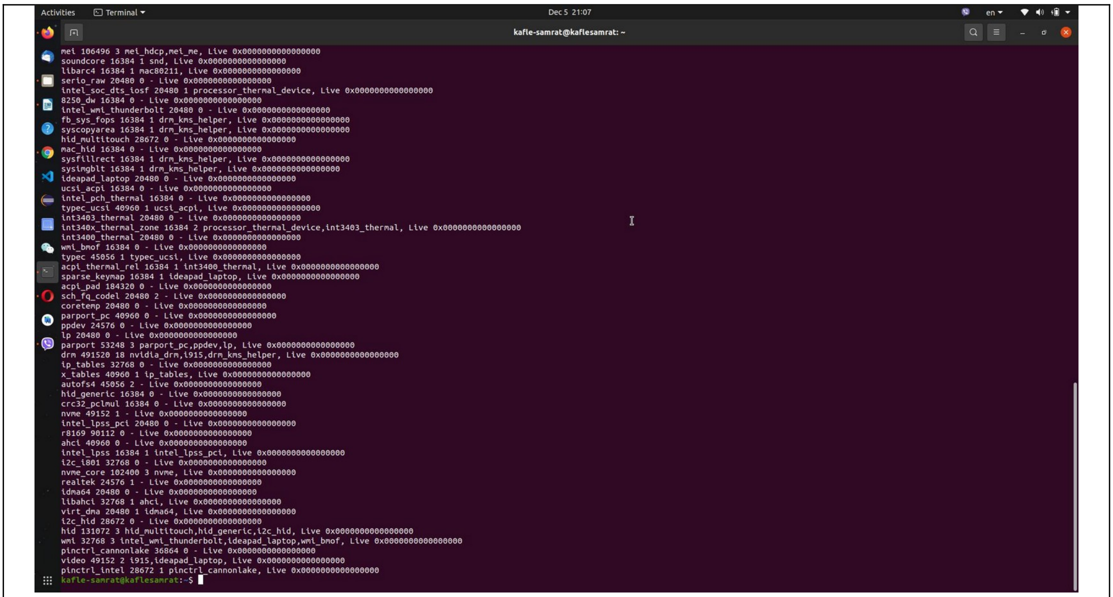


```
kafle-samrat@kaflesamrat:~$ cat /proc/filesystems
nodev   sysfs
nodev   tmpfs
nodev   bdev
nodev   proc
nodev   cgroup
nodev   cgroup2
nodev   cpuset
nodev   devtmpfs
nodev   configfs
nodev   debugfs
nodev   tracefs
nodev   securityfs
nodev   sockfs
nodev   bpf
nodev   pipefs
nodev   ramfs
nodev   hugetlbfs
nodev   devpts
        ext3
        ext2
        ext4
        squashfs
        vfat
nodev   ecryptfs
        fuseblk
nodev   fuse
nodev   fusectl
nodev   mqueue
nodev   pstore
nodev   autofs
nodev   binfmt_misc
kafle-samrat@kaflesamrat:~$
```

cat *proc*modules   check modules

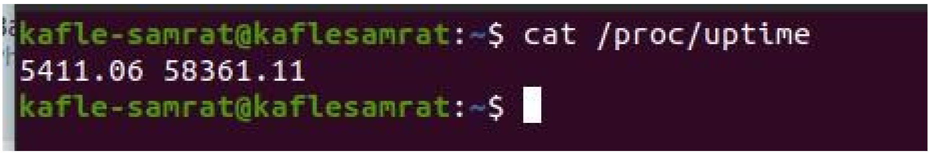


```
kafle-samrat@kaflesamrat:~$ cat /proc/modules
rfcomm 81920 4 - Live 0x0000000000000000
ccm 20480 6 - Live 0x0000000000000000
cmac 16384 3 - Live 0x0000000000000000
algif_hash 16384 1 - Live 0x0000000000000000
algif_skcipher 16384 1 - Live 0x0000000000000000
af_alg 24576 6 algif_hash,algif_skcipher, Live 0x0000000000000000
bnep 24576 2 - Live 0x0000000000000000
binfmt_misc 24576 1 - Live 0x0000000000000000
snd_hda_codec_hdmi 61440 2 - Live 0x0000000000000000
snd_hda_codec_realtek 126976 1 - Live 0x0000000000000000
snd_hda_codec_generic 81920 1 snd_hda_codec_realtek, Live 0x0000000000000000
nvidia_uvm 1007616 0 - Live 0x0000000000000000 (OE)
joydev 24576 0 - Live 0x0000000000000000
nvidia_drm 49152 14 - Live 0x0000000000000000 (POE)
nvidia_modeset 1183744 23 nvidia_drm, Live 0x0000000000000000 (POE)
mei_hdcp 24576 0 - Live 0x0000000000000000
intel_rapl_msr 20480 0 - Live 0x0000000000000000
snd_sof_pci 20480 0 - Live 0x0000000000000000
snd_sof_intel_hda_common 69632 1 snd_sof_pci, Live 0x0000000000000000
snd_soc_hdac_hda 24576 1 snd_sof_intel_hda_common, Live 0x0000000000000000
snd_sof_intel_hda 20480 1 snd_sof_intel_hda_common, Live 0x0000000000000000
snd_sof_intel_byt 20480 1 snd_sof_pci, Live 0x0000000000000000
snd_sof_intel_ipc 20480 1 snd_sof_intel_byt, Live 0x0000000000000000
snd_sof 106496 4 snd_sof_pci,snd_sof_intel_hda_common,snd_sof_intel_byt,snd_sof_intel_ipc, Live 0x0000000000000000
snd_sof_xtensa_dsp 16384 1 snd_sof_pci, Live 0x0000000000000000
snd_hda_ext_core 28672 3 snd_sof_intel_hda_common,snd_soc_hdac_hda,snd_sof_intel_hda, Live 0x0000000000000000
snd_soc_acpi_intel_match 32768 2 snd_sof_pci,snd_sof_intel_hda_common, Live 0x0000000000000000
snd_soc_acpi 16384 2 snd_sof_pci,snd_soc_acpi_intel_match, Live 0x0000000000000000
ledtrig_audio 16384 3 snd_hda_codec_realtek,snd_hda_codec_generic,snd_sof, Live 0x0000000000000000
snd_soc_core 245760 3 snd_sof_intel_hda_common,snd_soc_hdac_hda,snd_sof, Live 0x0000000000000000
snd_compress 24576 1 snd_soc_core, Live 0x0000000000000000
ac97_bus 16384 1 snd_soc_core, Live 0x0000000000000000
snd_pcm_dmaengine 16384 1 snd_soc_core, Live 0x0000000000000000
snd_hda_intel 53248 10 - Live 0x0000000000000000
snd_intel_dspcfg 24576 3 snd_sof_pci,snd_sof_intel_hda_common,snd_hda_intel, Live 0x0000000000000000
snd_hda_codec 135168 5 snd_hda_codec_hdmi,snd_hda_codec_realtek,snd_hda_codec_generic,snd_soc_hdac_hda,snd_hda_intel, Live 0x0000000000000000
snd_hda_core 90112 9 snd_hda_codec_hdmi,snd_hda_codec_realtek,snd_hda_codec_generic,snd_sof_intel_hda_common,snd_soc_hdac_hda,snd_sof_intel_hda,snd_hda_ext_core,snd_hda_intel,snd_hda_codec, Live 0x000000000
0000000
snd_hwdep 20480 1 snd_hda_codec, Live 0x0000000000000000
snd_pcm 106496 9 snd_hda_codec_hdmi,snd_sof_intel_hda_common,snd_sof,snd_soc_core,snd_pcm_dmaengine,snd_hda_intel,snd_hda_codec,snd_hda_core, Live 0x0000000000000000
x86_pkg_temp_thermal 20480 0 - Live 0x0000000000000000
intel_powerclamp 20480 0 - Live 0x0000000000000000
kvm_intel 282624 0 - Live 0x0000000000000000
rtwpci 24576 0 - Live 0x0000000000000000
snd_seq_midi 20480 0 - Live 0x0000000000000000
snd_seq_midi_event 16384 1 snd_seq_midi, Live 0x0000000000000000
rtw88 618496 1 rtwpci, Live 0x0000000000000000
nvidia 19722240 1096 nvidia_uvm,nvidia_modeset, Live 0x0000000000000000 (POE)
snd_rawmidi 36864 1 snd_seq_midi, Live 0x0000000000000000
kvm 663552 1 kvm_intel, Live 0x0000000000000000
mac80211 843776 2 rtwpci,rtw88, Live 0x0000000000000000
crct10dif_pclmul 16384 1 - Live 0x0000000000000000
snd_seq 69632 2 snd_seq_midi,snd_seq_midi_event, Live 0x0000000000000000
ghash_clmulni_intel 16384 0 - Live 0x0000000000000000
i915 1986560 4 - Live 0x0000000000000000
```



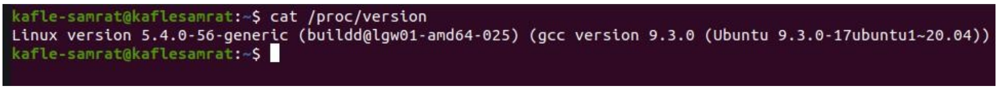
```
mei 106496 3 mei_hdcp,mei_me, Live 0x0000000000000000
soundcore 16384 1 snd, Live 0x0000000000000000
libarc4 16384 1 mac80211, Live 0x0000000000000000
serio_raw 20480 0 - Live 0x0000000000000000
intel_soc_dts_iosf 20480 1 processor_thermal_device, Live 0x0000000000000000
8250_dw 16384 0 - Live 0x0000000000000000
intel_wmi_thunderbolt 20480 0 - Live 0x0000000000000000
fb_sys_fops 16384 1 drm_kms_helper, Live 0x0000000000000000
syscopyarea 16384 1 drm_kms_helper, Live 0x0000000000000000
hid_multitouch 28672 0 - Live 0x0000000000000000
mac_hid 16384 0 - Live 0x0000000000000000
sysfillrect 16384 1 drm_kms_helper, Live 0x0000000000000000
sysimgblt 16384 1 drm_kms_helper, Live 0x0000000000000000
ideapad_laptop 20480 0 - Live 0x0000000000000000
ucsi_acpi 16384 0 - Live 0x0000000000000000
intel_pch_thermal 16384 0 - Live 0x0000000000000000
typec_ucsi 40960 1 ucsi_acpi, Live 0x0000000000000000
int3403_thermal 20480 0 - Live 0x0000000000000000
int340x_thermal_zone 16384 2 processor_thermal_device,int3403_thermal, Live 0x0000000000000000
int3400_thermal 20480 0 - Live 0x0000000000000000
wmi_bmof 16384 0 - Live 0x0000000000000000
typec 45056 1 typec_ucsi, Live 0x0000000000000000
acpi_thermal_rel 16384 1 int3400_thermal, Live 0x0000000000000000
sparse_keymap 16384 1 ideapad_laptop, Live 0x0000000000000000
acpi_pad 184320 0 - Live 0x0000000000000000
sch_fq_codel 20480 2 - Live 0x0000000000000000
coretemp 20480 0 - Live 0x0000000000000000
parport_pc 40960 0 - Live 0x0000000000000000
ppdev 24576 0 - Live 0x0000000000000000
lp 20480 0 - Live 0x0000000000000000
parport 53248 3 parport_pc,ppdev,lp, Live 0x0000000000000000
drm 491520 18 nvidia_drm,i915,drm_kms_helper, Live 0x0000000000000000
ip_tables 32768 0 - Live 0x0000000000000000
x_tables 40960 1 ip_tables, Live 0x0000000000000000
autofs4 45056 2 - Live 0x0000000000000000
hid_generic 16384 0 - Live 0x0000000000000000
crc32_pclmul 16384 0 - Live 0x0000000000000000
nvme 49152 1 - Live 0x0000000000000000
intel_lpss_pci 20480 0 - Live 0x0000000000000000
r8169 90112 0 - Live 0x0000000000000000
ahci 40960 0 - Live 0x0000000000000000
intel_lpss 16384 1 intel_lpss_pci, Live 0x0000000000000000
i2c_i801 32768 0 - Live 0x0000000000000000
nvme_core 102400 3 nvme, Live 0x0000000000000000
realtek 24576 1 - Live 0x0000000000000000
idma64 20480 0 - Live 0x0000000000000000
libahci 32768 1 ahci, Live 0x0000000000000000
virt_dma 20480 1 idma64, Live 0x0000000000000000
i2c_hid 28672 0 - Live 0x0000000000000000
hid 131072 3 hid_multitouch,hid_generic,i2c_hid, Live 0x0000000000000000
wmi 32768 3 intel_wmi_thunderbolt,ideapad_laptop,wmi_bmof, Live 0x0000000000000000
pinctrl_cannonlake 36864 0 - Live 0x0000000000000000
video 49152 2 i915,ideapad_laptop, Live 0x0000000000000000
pinctrl_intel 28672 1 pinctrl_cannonlake, Live 0x0000000000000000
kafle-samrat@kaflesamrat:~$
```

cat proc/uptime   shows time



```
kafle-samrat@kaflesamrat:~$ cat /proc/uptime
5411.06 58361.11
kafle-samrat@kaflesamrat:~$
```

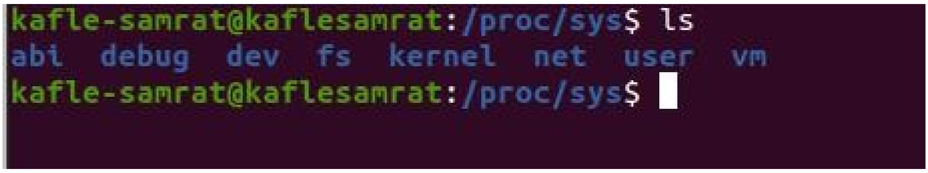
cat *proc/*version   check version



```
kafle-samrat@kaflesamrat:~$ cat /proc/version
Linux version 5.4.0-56-generic (buildd@lgw01-amd64-025) (gcc version 9.3.0 (Ubuntu 9.3.0-17ubuntu1~20.04))
kafle-samrat@kaflesamrat:~$
```

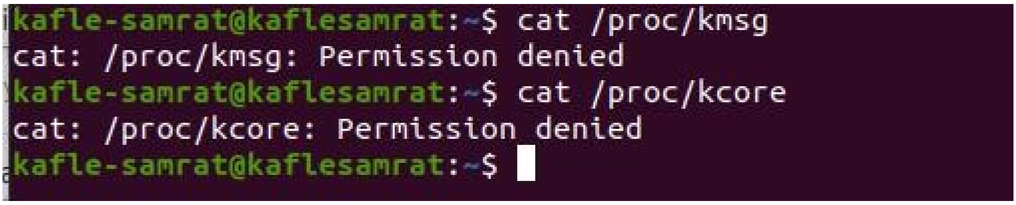
la *proc*/sys   list system files

.



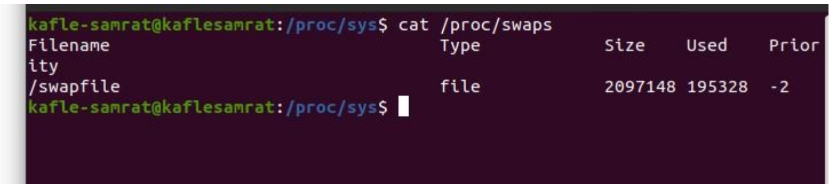
```
kafle-samrat@kaflesamrat:/proc/sys$ ls
abi  debug  dev  fs  kernel  net  user  vm
kafle-samrat@kaflesamrat:/proc/sys$
```

cat proc/kmsg



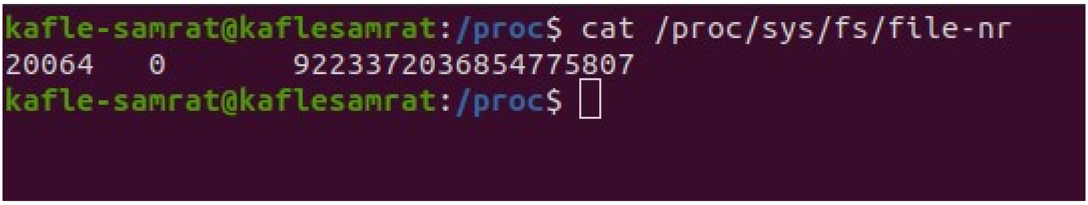
```
kafle-samrat@kaflesamrat:~$ cat /proc/kmsg
cat: /proc/kmsg: Permission denied
kafle-samrat@kaflesamrat:~$ cat /proc/kcore
cat: /proc/kcore: Permission denied
kafle-samrat@kaflesamrat:~$
```

cat *proc/*swaps



```
kafle-samrat@kaflesamrat:/proc/sys$ cat /proc/swaps
Filename                                Type            Size    Used    Prior
ity
/swapfile                               file            2097148 195328  -2
kafle-samrat@kaflesamrat:/proc/sys$
```

Cat /proc/sys/fs/file-nr looks at the current usage of the file handle

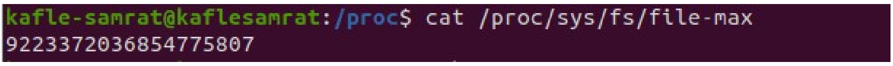


```
kafle-samrat@kaflesamrat:/proc$ cat /proc/sys/fs/file-nr
20064   0       9223372036854775807
kafle-samrat@kaflesamrat:/proc$
```

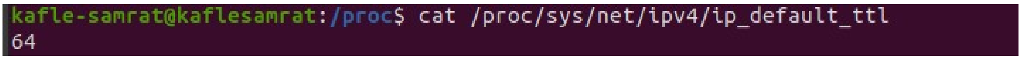
.

The total number of file handles allocated, the number of file handles currently used, and the maximum number of file handles that can be allocated.

.

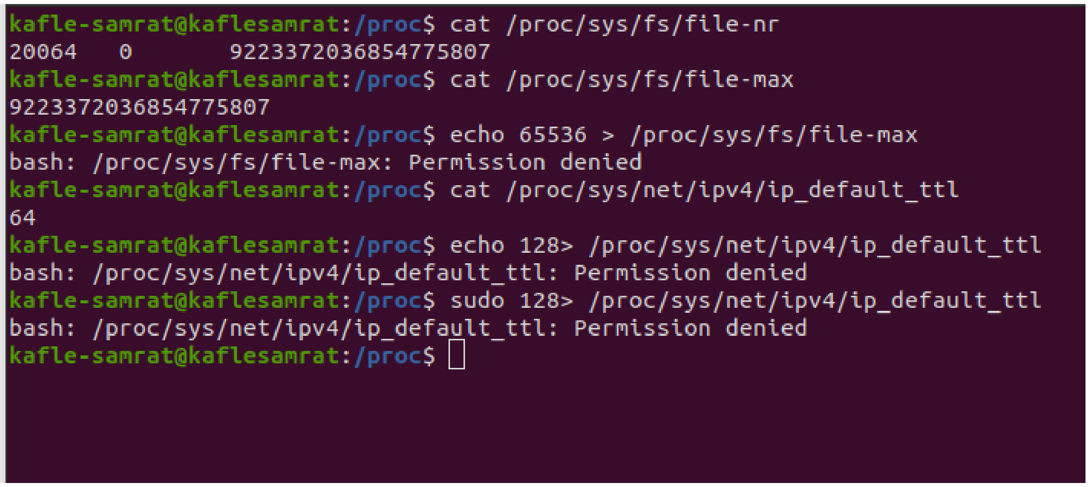


```
kafle-samrat@kaflesamrat:/proc$ cat /proc/sys/fs/file-max
9223372036854775807
```



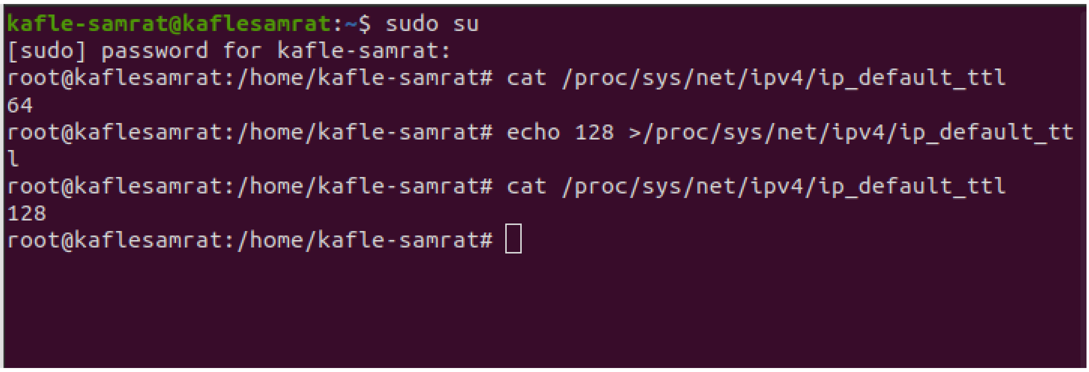
```
kafle-samrat@kaflesamrat:/proc$ cat /proc/sys/net/ipv4/ip_default_ttl
64
```

.



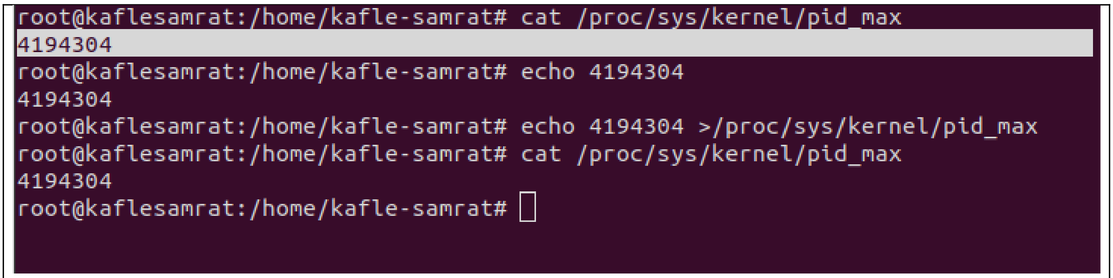
```
kafle-samrat@kaflesamrat:/proc$ cat /proc/sys/fs/file-nr
20064   0       9223372036854775807
kafle-samrat@kaflesamrat:/proc$ cat /proc/sys/fs/file-max
9223372036854775807
kafle-samrat@kaflesamrat:/proc$ echo 65536 > /proc/sys/fs/file-max
bash: /proc/sys/fs/file-max: Permission denied
kafle-samrat@kaflesamrat:/proc$ cat /proc/sys/net/ipv4/ip_default_ttl
64
kafle-samrat@kaflesamrat:/proc$ echo 128> /proc/sys/net/ipv4/ip_default_ttl
bash: /proc/sys/net/ipv4/ip_default_ttl: Permission denied
kafle-samrat@kaflesamrat:/proc$ sudo 128> /proc/sys/net/ipv4/ip_default_ttl
bash: /proc/sys/net/ipv4/ip_default_ttl: Permission denied
kafle-samrat@kaflesamrat:/proc$ 
```

.



```
kafle-samrat@kaflesamrat:~$ sudo su
[sudo] password for kafle-samrat:
root@kaflesamrat:/home/kafle-samrat# cat /proc/sys/net/ipv4/ip_default_ttl
64
root@kaflesamrat:/home/kafle-samrat# echo 128 >/proc/sys/net/ipv4/ip_default_tt
l
root@kaflesamrat:/home/kafle-samrat# cat /proc/sys/net/ipv4/ip_default_ttl
128
root@kaflesamrat:/home/kafle-samrat# 
```

.



```
root@kaflesamrat:/home/kafle-samrat# cat /proc/sys/kernel/pid_max
4194304
root@kaflesamrat:/home/kafle-samrat# echo 4194304
4194304
root@kaflesamrat:/home/kafle-samrat# echo 4194304 >/proc/sys/kernel/pid_max
root@kaflesamrat:/home/kafle-samrat# cat /proc/sys/kernel/pid_max
4194304
root@kaflesamrat:/home/kafle-samrat# 
```

.

实验心得机会:

通过这次实验我了解到 PROC 文件系统是一个虚拟的文件系统，通过文件系统的接口实现，用于输出系统的运行状态。形式的文件系统,它提供了一个接口,用于操作系统本身和应用程序之间的通信过程,这样应用程序就可以安全地和容易获得的当前健康系统和内核的内部数据信息,并且可以修改某些系统的配置信息。此外，由于 PROC 是作为文件系统接口实现的，所以用户可以像访问常规文件一样访问它，但是它只存在于内存中，而不存在于实际的物理磁盘上。因此，当系统重新启动并关闭电源时，系统中的所有数据和信息都会消失。

实验报告

组别		**姓名**	KAFLE SAMRAT	**同组实验者**	
实验项目名称	Linux驱动程序			**实验日期**	月 日
教师评语					
实验成绩:			**指导教师（签名）：** 2020年 月 日		

一．实验目的

在linux系统中，一个硬件设备想要运行同样需要提供设备驱动程序，底层的原理和MCU中的设备驱动程序一样：收发数据以及处理数据，只是由于桌面操作系统的特殊性，设备驱动程序的流程会复杂很多。

linux将内核与用户分离，驱动模块运行在内核空间中，而应用程序运行在用户空间，内核主要对公共且有限的资源进行管理、调度，比如硬件外设资源、内存资源等。

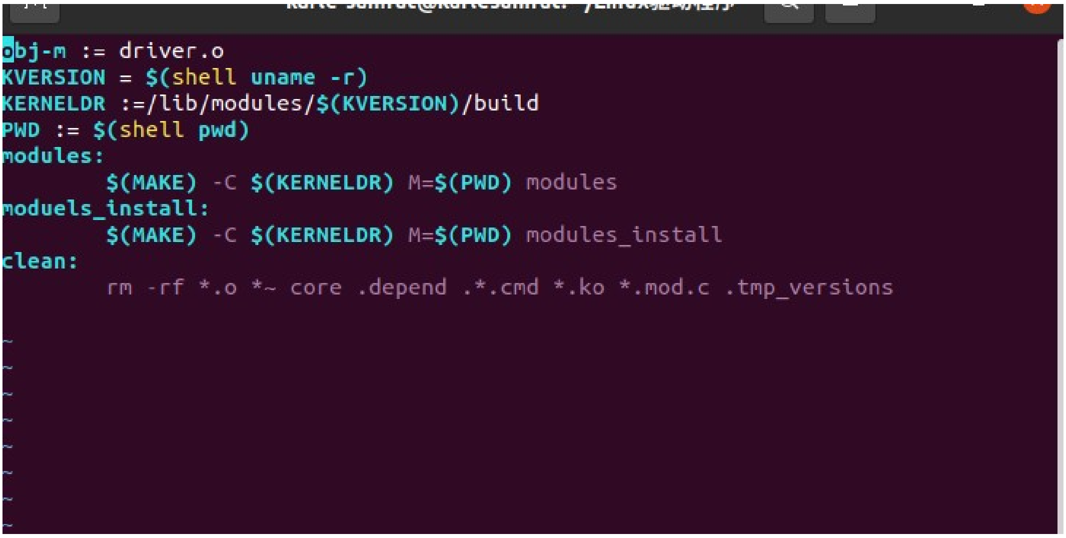
当用户需要使用系统资源时，通过系统调用进入内核，由内核基于某种调度算法对这部分资源进行调度。

二．实验内容

从教材提供的电子资源中找到或者按教材提示自己编写简单的Linux内核模块driver.c及其对应的Makefile文件

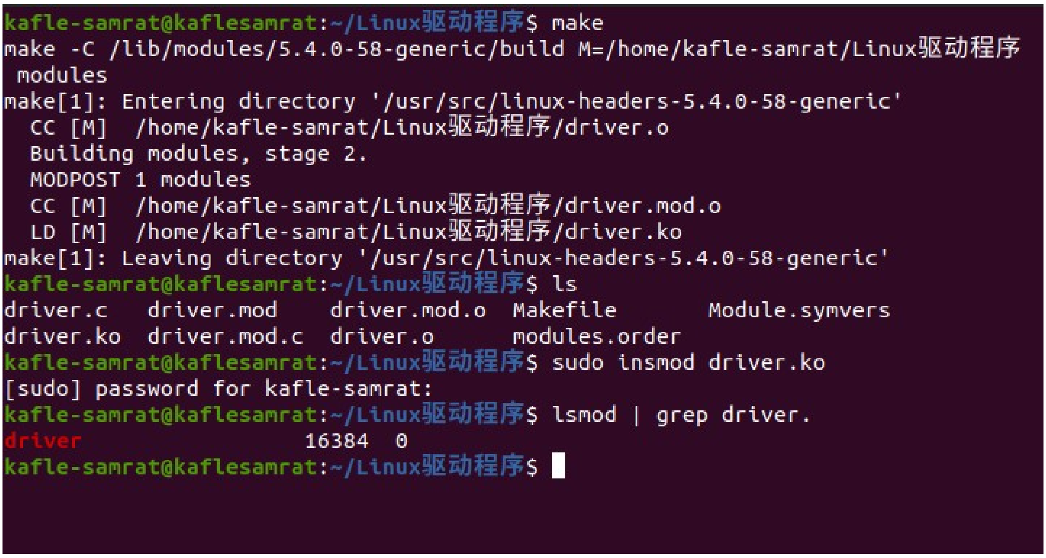
```
#include <linux/init.h>
#include <linux/module.h>
#include <linux/kernel.h>
#include <linux/kthread.h>
#include <linux/delay.h>
#include <linux/kobject.h>
#include <linux/sysfs.h>
#include <linux/slab.h>
#include <linux/string.h>
#include <linux/gpio.h>

MODULE_LICENSE("GPL");
```

```
MODULE_AUTHOR("Downey");
MODULE_DESCRIPTION("Kobject test!");
MODULE_VERSION("0.1");

static int led_status = 0;
#define LED_PIN   26

static struct kobject *kob;

static ssize_t led_show(struct kobject*
kobjs,struct kobj_attribute *attr,char *buf)
{
    printk(KERN_INFO "Read led\n");
    return sprintf(buf,"The led_status status =
%d\n",led_status);
}

static ssize_t led_status_show(struct kobject*
kobjs,struct kobj_attribute *attr,char *buf)
{
    printk(KERN_INFO "led status show\n");
    return sprintf(buf,"led status : \n%d\
n",led_status);
}

static ssize_t led_status_store(struct kobject
*kobj, struct kobj_attribute *attr,const char
*buf, size_t count)
{
    printk(KERN_INFO "led status store\n");
    if(0 == memcmp(buf,"on",2))
    {
        gpio_set_value(LED_PIN,1);
        led_status = 1;
    }
    else if(0 == memcmp(buf,"off",3))
    {
        gpio_set_value(LED_PIN,0);
        led_status = 0;
    }
    else
    {
        printk(KERN_INFO "Not support cmd\n");
    }

    return count;
}

static struct kobj_attribute status_attr =
__ATTR_RO(led);
static struct kobj_attribute led_attr =
__ATTR(led_status,0660,led_status_show,led_status_store);  //Doesn't support 0666 in new version.

static struct attribute *led_attrs[] = {
    &status_attr.attr,
    &led_attr.attr,
    NULL,
};

static struct attribute_group attr_g = {
    .name = "kobject_test",
    .attrs = led_attrs,
};


int create_kobject(void)
{
    kob = kobject_create_and_add("obj_test",kernel_kobj->parent);
    return 0;
}

static void gpio_config(void)
{
    if(!gpio_is_valid(LED_PIN)){
        printk(KERN_ALERT "Error wrong gpio
number\n");
        return ;
    }
    gpio_request(LED_PIN,"led_ctr");
    gpio_direction_output(LED_PIN,1);
    gpio_set_value(LED_PIN,1);
    led_status = 1;
}

static void gpio_deconfig(void)
{
    gpio_free(LED_PIN);
}

static int __init sysfs_ctrl_init(void){
    printk(KERN_INFO "Kobject test!\n");
    gpio_config();
    create_kobject();

    sysfs_create_group(kob, &attr_g);

    return 0;

}


static void __exit sysfs_ctrl_exit(void){


    gpio_deconfig();

    kobject_put(kob);

    printk(KERN_INFO "Goodbye!\n");

}


module_init(sysfs_ctrl_init);

module_exit(sysfs_ctrl_exit);
```



```
obj-m := driver.o
KVERSION = $(shell uname -r)
KERNELDR :=/lib/modules/$(KVERSION)/build
PWD := $(shell pwd)
modules:
        $(MAKE) -C $(KERNELDR) M=$(PWD) modules
moduels_install:
        $(MAKE) -C $(KERNELDR) M=$(PWD) modules_install
clean:
        rm -rf *.o *~ core .depend .*.cmd *.ko *.mod.c .tmp_versions

~
~
~
~
~
~
~
~
```

```
obj-m := driver.o

KVERSION = $(shell uname -r)

KERNELDR :=/lib/modules/$(KVERSION)/build

PWD := $(shell pwd)

modules:

   $(MAKE) -C $(KERNELDR) M=$(PWD) modules

moduels_install:

   $(MAKE) -C $(KERNELDR) M=$(PWD) modules_install

clean:
    rm -rf *.o *~ core .depend .*.cmd *.ko *.mod.c .tmp_versions
```



```
kafle-samrat@kaflesamrat:~/Linux驱动程序$ make
make -C /lib/modules/5.4.0-58-generic/build M=/home/kafle-samrat/Linux驱动程序
 modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-58-generic'
  CC [M]  /home/kafle-samrat/Linux驱动程序/driver.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/kafle-samrat/Linux驱动程序/driver.mod.o
  LD [M]  /home/kafle-samrat/Linux驱动程序/driver.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-58-generic'
kafle-samrat@kaflesamrat:~/Linux驱动程序$ ls
driver.c   driver.mod    driver.mod.o  Makefile       Module.symvers
driver.ko  driver.mod.c  driver.o      modules.order
kafle-samrat@kaflesamrat:~/Linux驱动程序$ sudo insmod driver.ko
[sudo] password for kafle-samrat:
kafle-samrat@kaflesamrat:~/Linux驱动程序$ lsmod | grep driver.
driver                 16384  0
kafle-samrat@kaflesamrat:~/Linux驱动程序$ 
```

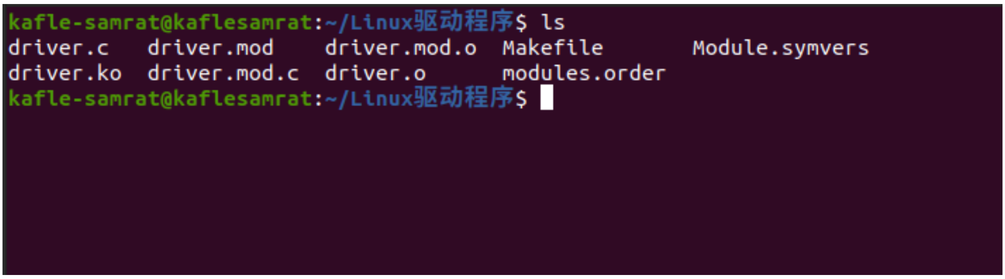
编译、安装、删除该模块，查看该模块的安装位置、运行情况

本次采用单独编译、动态插入内核；把将开发的内核代码文件直接进行编译，然后使用命令动态插入内核或者从内核卸载。

优点：编译速度快；单独调试代码

缺点：每次系统启动后都需要再加载代码

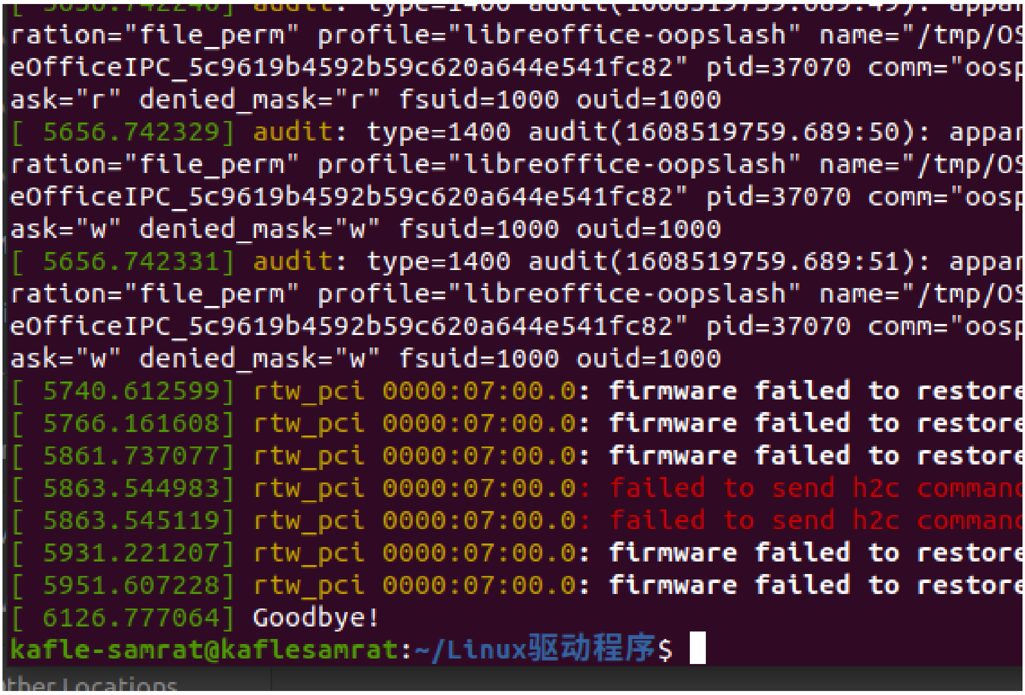
.



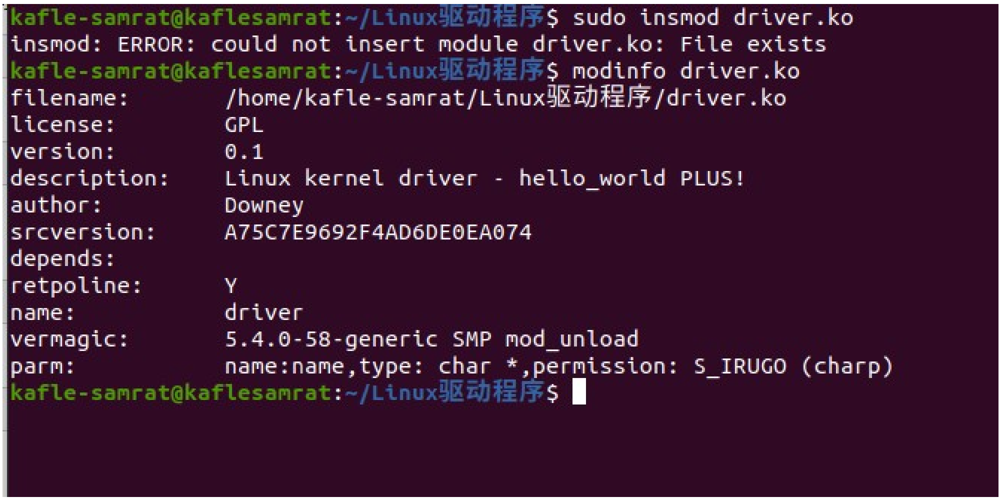
```
kafle-samrat@kaflesamrat:~/Linux驱动程序$ ls
driver.c    driver.mod     driver.mod.o  Makefile       Module.symvers
driver.ko   driver.mod.c   driver.o      modules.order
kafle-samrat@kaflesamrat:~/Linux驱动程序$ 
```

删除模块：

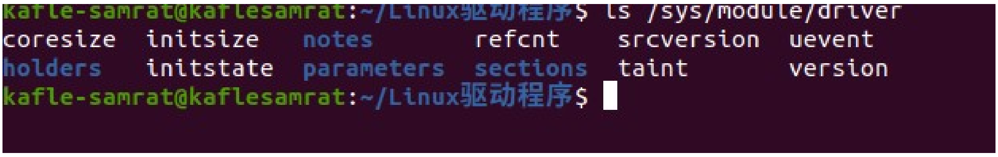
```
sudo rmmod driver.ko
```



```
[ 5656.742240] audit: type=1400 audit(1608519759.689:49): appar
ration="file_perm" profile="libreoffice-oopslash" name="/tmp/OS
eOfficeIPC_5c9619b4592b59c620a644e541fc82" pid=37070 comm="oosp
ask="r" denied_mask="r" fsuid=1000 ouid=1000
[ 5656.742329] audit: type=1400 audit(1608519759.689:50): appar
ration="file_perm" profile="libreoffice-oopslash" name="/tmp/OS
eOfficeIPC_5c9619b4592b59c620a644e541fc82" pid=37070 comm="oosp
ask="w" denied_mask="w" fsuid=1000 ouid=1000
[ 5656.742331] audit: type=1400 audit(1608519759.689:51): appar
ration="file_perm" profile="libreoffice-oopslash" name="/tmp/OS
eOfficeIPC_5c9619b4592b59c620a644e541fc82" pid=37070 comm="oosp
ask="w" denied_mask="w" fsuid=1000 ouid=1000
[ 5740.612599] rtw_pci 0000:07:00.0: firmware failed to restore
[ 5766.161608] rtw_pci 0000:07:00.0: firmware failed to restore
[ 5861.737077] rtw_pci 0000:07:00.0: firmware failed to restore
[ 5863.544983] rtw_pci 0000:07:00.0: failed to send h2c command
[ 5863.545119] rtw_pci 0000:07:00.0: failed to send h2c command
[ 5931.221207] rtw_pci 0000:07:00.0: firmware failed to restore
[ 5951.607228] rtw_pci 0000:07:00.0: firmware failed to restore
[ 6126.777064] Goodbye!
kafle-samrat@kaflesamrat:~/Linux驱动程序$ 
```



```
kafle-samrat@kaflesamrat:~/Linux驱动程序$ sudo insmod driver.ko
insmod: ERROR: could not insert module driver.ko: File exists
kafle-samrat@kaflesamrat:~/Linux驱动程序$ modinfo driver.ko
filename:       /home/kafle-samrat/Linux驱动程序/driver.ko
license:        GPL
version:        0.1
description:    Linux kernel driver - hello_world PLUS!
author:         Downey
srcversion:     A75C7E9692F4AD6DE0EA074
depends:        
retpoline:      Y
name:           driver
vermagic:       5.4.0-58-generic SMP mod_unload 
parm:           name:name,type: char *,permission: S_IRUGO (charp)
kafle-samrat@kaflesamrat:~/Linux驱动程序$ 
```



```
kafle-samrat@kaflesamrat:~/Linux驱动程序$ ls /sys/module/driver
coresize  initsize   notes        refcnt    srcversion  uevent
holders   initstate  parameters  sections  taint       version
kafle-samrat@kaflesamrat:~/Linux驱动程序$ 
```

.

.

.

实验心得:

通过这次实验，我学习了基本的Linux内核模块开发框架和编译方法，熟悉了添加Linux内核模块的过程，了解了程序的工作原理。

实 验 报 告

组 别		姓 名	KAFLE SAMRAT	同组实验者	
实验项目名称	进程同步：生产者/消费者问题			实验日期	月 日
教师评语					
实验成绩:			指导教师（签名）： 2020年 月 日		

一．实验目的

在系统中有 b 个缓冲区（每个可以放 1 个产品）构成的仓库。有 n 个生产者 $p_1,p_2,\cdots,p_n$，每个生产者 $p_i(i=1,2,\cdots,n)$ 可以生产 k_i 个产品。有 m 个消费者 $c_1,c_2,\cdots,c_m$，每个消费者消费若干个产品，直到消费完所有的产品为止，即 $\sum_{i=1}^{n}k_i=\sum_{j=1}^{m}q_j$，其中 $q_j(j=1,2,\cdots,m)$ 是消费者 c_j 实际消费的产品个数。

输入：生产者个数 n、消费者个数 m、缓冲区个数 b、每个生产者生产产品的个数 $k_i(i=1,2,\cdots,n)$ 等。

输出：生产者-消费者并发执行的过程、每个进程的状态变化。如：

（1）生产者 p_i 将第 X 个产品放入仓库；

（2）消费者 c_j 从仓库中消费第 Y 个产品；

（3）生产者 p_i 阻塞；

（4）消费者 c_j 阻塞；

（5）生产者 p_i 被唤醒；

（6）消费者 c_j 被唤醒。

二．实验内容

1.通过 pthread_t 来创建线程对象，通过 pthread_create 来实现对线程的执行创建。

2.创建 linuxlab.c 文件来实现对互斥的实现。

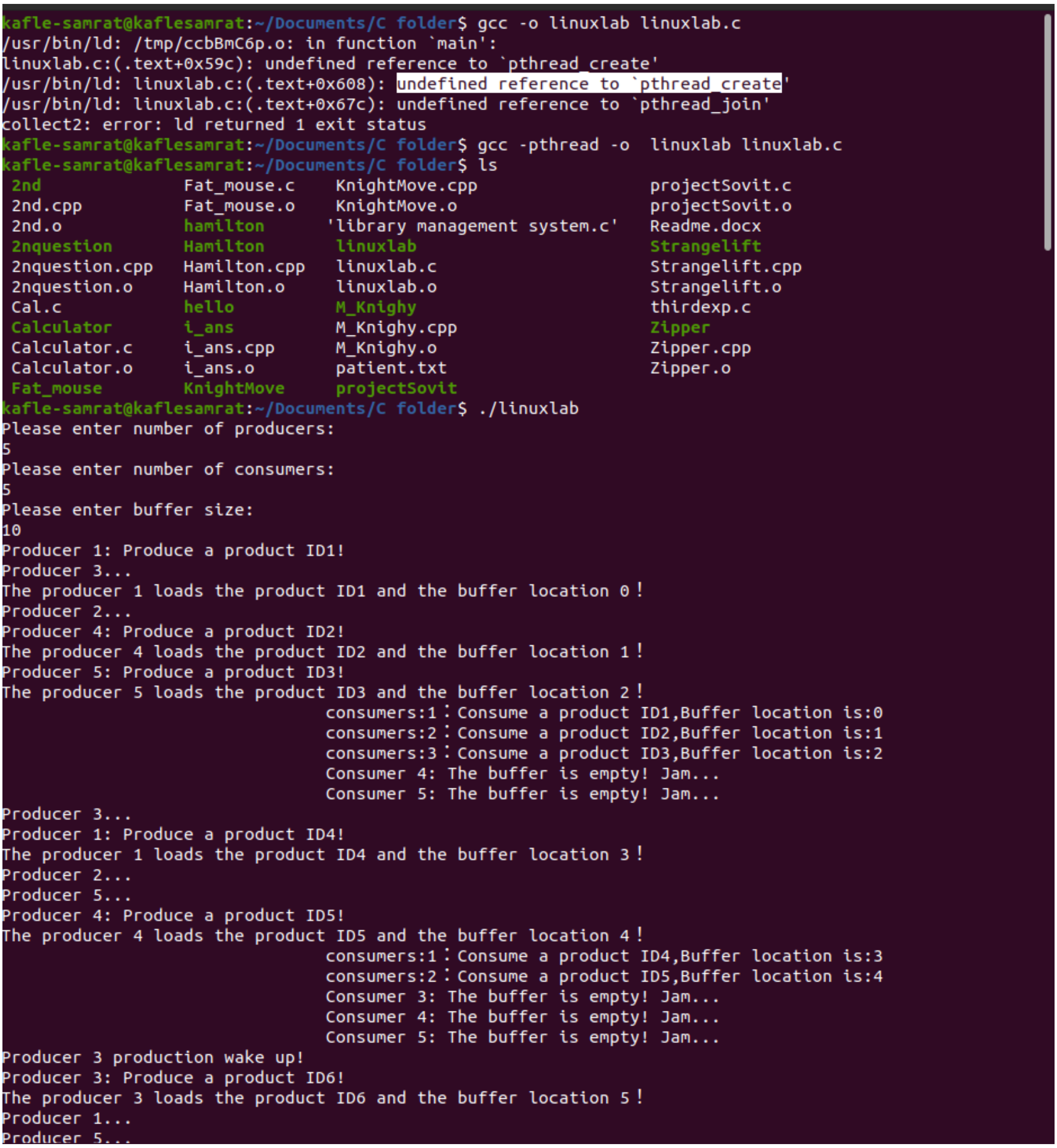
code：

```
#include <stdio.h>
#include <pthread.h>
#include <unistd.h>
#include <stdlib.h>
#define true 1

int product_id = 0;

int consumer_id = 0;

int N;
int producerNum;

int consumerNum;

typedef int semaphore;

typedef int item;

item* buffer;

int in = 0;

int out = 0;

int proCount = 0;

semaphore mutex = 1, empty , full = 0, proCmutex = 1;

void * producer(void * a){
    int id = ++product_id;
    while(true){
        int flag = 0;

        while(empty <= 0){
            printf("Producer %d: Buffer full! jam……\n",id);
            flag =1;
            sleep(1);
        }
        if(flag == 1)
            printf("Producer %d wakes up with empty buffer!\n",id);

        flag = 0;
        while(proCmutex <= 0){printf("Producer %d...\n",id);flag =
1;sleep(1);};
```

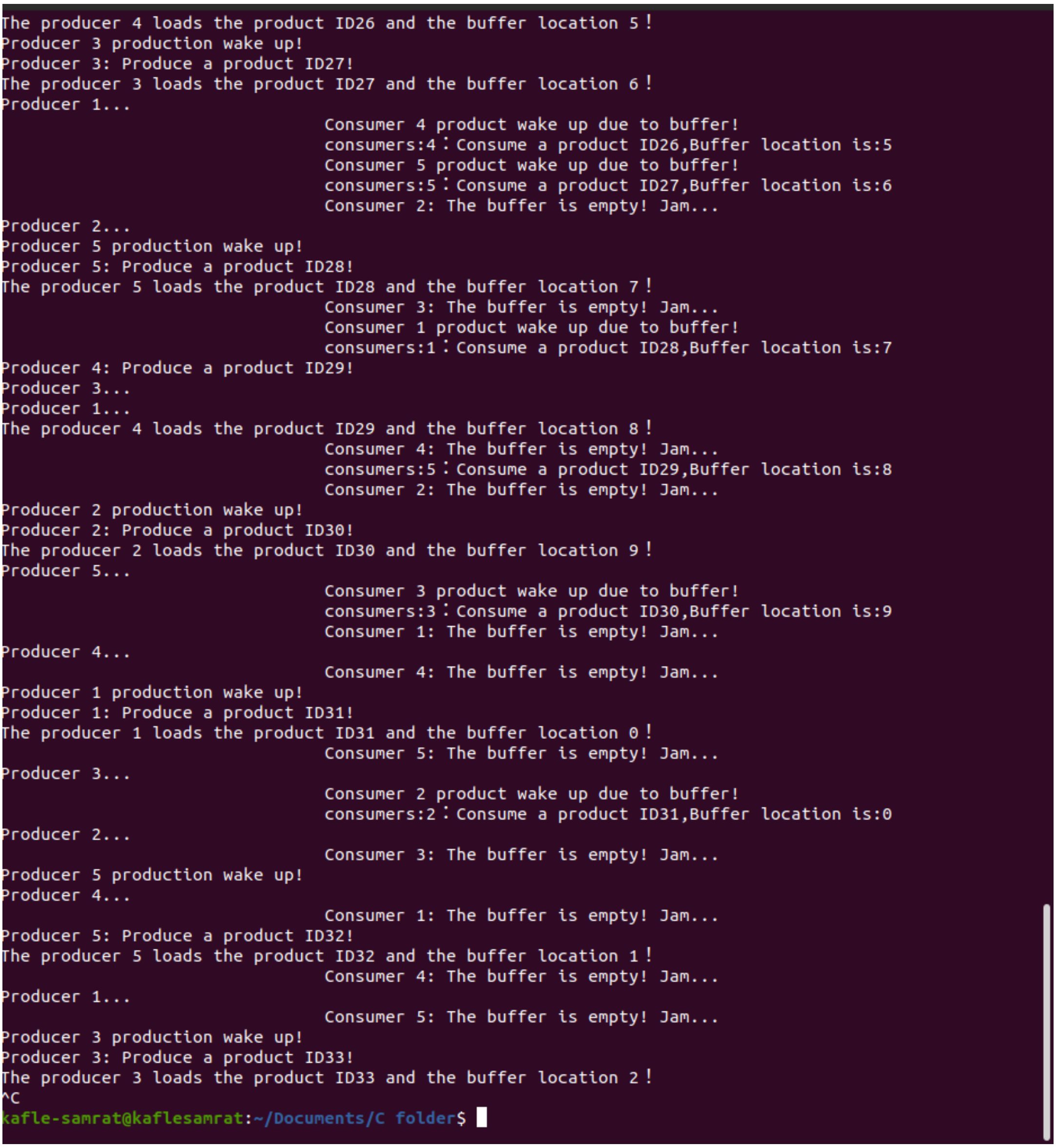
```
        proCmutex--;
        if(flag == 1)
            printf("Producer %d production wake up!\n",id);

        proCount++;
        printf("Producer %d: Produce a product ID%d!\
n",id,proCount);

        flag = 0;
        while(mutex <= 0){printf("Producer %d loaded into the
block......\n",id);sleep(1);flag=1;};

        mutex--;

        if(flag == 1)
            printf("The producer %d load wakes up, loads the product

ID%d, and the buffer location %d！ \n",id,proCount,in);
        else
            printf("The producer %d loads the product ID%d and the

buffer location %d！ \n",id,proCount,in);


        empty--;

        buffer[in] = proCount;

        in = (in + 1) % N;



        full++;

        mutex++;

        proCmutex++;
        sleep(1);
    }
}

void * consumer(void *b){
    int id = ++consumer_id;
    while(true){
        int flag = 0;
        while(full <= 0){
            printf("\t\t\t\tConsumer %d: The buffer is empty! Jam...\
n",id);
            flag = 1;
            sleep(1);
        }
        full--;
        if(flag ==1)
            printf("\t\t\t\tConsumer %d product wake up due to buffer!\
n",id);
        flag = 0;

        while(mutex <= 0){printf("\t\t\t\tIn consumer %d
consumption congestion... …\n",id);sleep(1);};
        mutex--;
        if(flag == 1)

            printf("\t\t\t\tConsumer %d consumption wake up！\n",id);

        int nextc = buffer[out];
        buffer[out] = 0;//Set the buffer to 0 after consumption
        empty++;

        printf("\t\t\t\tconsumers:%d：Consume a product ID%d,Buffer

location is:%d\n",id, nextc,out);
        out = (out + 1) % N;
        mutex++;
        sleep(1);
    }
```

```
}

int main()
{
    int tempnum;

    printf("Please enter number of producers:\n");
    scanf("%d",&tempnum);
    producerNum = tempnum;

    printf("Please enter number of consumers:\n");
    scanf("%d",&tempnum);
    consumerNum = tempnum;

    printf("Please enter buffer size:\n");
    scanf("%d",&tempnum);
    N = tempnum;
    empty = N;
    buffer = (item*)malloc(N*sizeof(item));
    for(int i=0;i<N;i++)
    {
        buffer[i]=0;
    }

    pthread_t threadPool[producerNum+consumerNum];//Declares
an array of threads as a thread pool
    int i;
    for(i = 0; i < producerNum; i++){
        pthread_t temp;
        //In an if statement, the first argument is a thread pointer,
the second is a thread property pointer, and the third is a function
pointer, that is, what the thread is going to execute code
        //Functions create objects via the producer pointer and
assign values to TEMp for execution as threads
        if(pthread_create(&temp, NULL, producer, NULL) == -1)
        {
            printf("ERROR, fail to create producer%d\n", i);
            exit(1);
        }
        //Temp is put into the process pool as a thread that can
execute
        threadPool[i] = temp;
    }//Create a producer process to put into the thread pool

    //Processes are also created for consumer processes
    for(i = 0; i < consumerNum; i++){
        pthread_t temp;
        if(pthread_create(&temp, NULL, consumer, NULL) == -1){
            printf("ERROR, fail to create consumer%d\n", i);
            exit(1);
        }
        threadPool[i+producerNum] = temp;
    }



    void * result;
    for(i = 0; i < producerNum+consumerNum; i++){

        if(pthread_join(threadPool[i], &result) == -1){
            printf("fail to recollect\n");
            exit(1);
        }
    }
    return 0;
}
```

实验结果：



```
kafle-samrat@kaflesamrat:~/Documents/C folder$ gcc -o linuxlab linuxlab.c
/usr/bin/ld: /tmp/ccbBmC6p.o: in function `main':
linuxlab.c:(.text+0x59c): undefined reference to `pthread_create'
/usr/bin/ld: linuxlab.c:(.text+0x608): undefined reference to `pthread_create'
/usr/bin/ld: linuxlab.c:(.text+0x67c): undefined reference to `pthread_join'
collect2: error: ld returned 1 exit status
kafle-samrat@kaflesamrat:~/Documents/C folder$ gcc -pthread -o  linuxlab linuxlab.c
kafle-samrat@kaflesamrat:~/Documents/C folder$ ls
 2nd              Fat_mouse.c    KnightMove.cpp                  projectSovit.c
 2nd.cpp          Fat_mouse.o    KnightMove.o                    projectSovit.o
 2nd.o            hamilton      'library management system.c'    Readme.docx
 2nquestion       Hamilton       linuxlab                        Strangelift
 2nquestion.cpp   Hamilton.cpp   linuxlab.c                      Strangelift.cpp
 2nquestion.o     Hamilton.o     linuxlab.o                      Strangelift.o
 Cal.c            hello          M_Knighy                        thirdexp.c
 Calculator       i_ans          M_Knighy.cpp                    Zipper
 Calculator.c     i_ans.cpp      M_Knighy.o                      Zipper.cpp
 Calculator.o     i_ans.o        patient.txt                     Zipper.o
 Fat_mouse        KnightMove     projectSovit
kafle-samrat@kaflesamrat:~/Documents/C folder$ ./linuxlab
Please enter number of producers:
5
Please enter number of consumers:
5
Please enter buffer size:
10
Producer 1: Produce a product ID1!
Producer 3...
The producer 1 loads the product ID1 and the buffer location 0！
Producer 2...
Producer 4: Produce a product ID2!
The producer 4 loads the product ID2 and the buffer location 1！
Producer 5: Produce a product ID3!
The producer 5 loads the product ID3 and the buffer location 2！
                                consumers:1：Consume a product ID1,Buffer location is:0
                                consumers:2：Consume a product ID2,Buffer location is:1
                                consumers:3：Consume a product ID3,Buffer location is:2
                                Consumer 4: The buffer is empty! Jam...
                                Consumer 5: The buffer is empty! Jam...
Producer 3...
Producer 1: Produce a product ID4!
The producer 1 loads the product ID4 and the buffer location 3！
Producer 2...
Producer 5...
Producer 4: Produce a product ID5!
The producer 4 loads the product ID5 and the buffer location 4！
                                consumers:1：Consume a product ID4,Buffer location is:3
                                consumers:2：Consume a product ID5,Buffer location is:4
                                Consumer 3: The buffer is empty! Jam...
                                Consumer 4: The buffer is empty! Jam...
                                Consumer 5: The buffer is empty! Jam...
Producer 3 production wake up!
Producer 3: Produce a product ID6!
The producer 3 loads the product ID6 and the buffer location 5！
Producer 1...
Producer 5...
```



```
The producer 4 loads the product ID26 and the buffer location 5！
Producer 3 production wake up!
Producer 3: Produce a product ID27!
The producer 3 loads the product ID27 and the buffer location 6！
Producer 1...
                                Consumer 4 product wake up due to buffer!
                                consumers:4：Consume a product ID26,Buffer location is:5
                                Consumer 5 product wake up due to buffer!
                                consumers:5：Consume a product ID27,Buffer location is:6
                                Consumer 2: The buffer is empty! Jam...
Producer 2...
Producer 5 production wake up!
Producer 5: Produce a product ID28!
The producer 5 loads the product ID28 and the buffer location 7！
                                Consumer 3: The buffer is empty! Jam...
                                Consumer 1 product wake up due to buffer!
                                consumers:1：Consume a product ID28,Buffer location is:7
Producer 4: Produce a product ID29!
Producer 3...
Producer 1...
The producer 4 loads the product ID29 and the buffer location 8！
                                Consumer 4: The buffer is empty! Jam...
                                consumers:5：Consume a product ID29,Buffer location is:8
                                Consumer 2: The buffer is empty! Jam...
Producer 2 production wake up!
Producer 2: Produce a product ID30!
The producer 2 loads the product ID30 and the buffer location 9！
Producer 5...
                                Consumer 3 product wake up due to buffer!
                                consumers:3：Consume a product ID30,Buffer location is:9
                                Consumer 1: The buffer is empty! Jam...
Producer 4...
                                Consumer 4: The buffer is empty! Jam...
Producer 1 production wake up!
Producer 1: Produce a product ID31!
The producer 1 loads the product ID31 and the buffer location 0！
                                Consumer 5: The buffer is empty! Jam...
Producer 3...
                                Consumer 2 product wake up due to buffer!
                                consumers:2：Consume a product ID31,Buffer location is:0
Producer 2...
                                Consumer 3: The buffer is empty! Jam...
Producer 5 production wake up!
Producer 4...
                                Consumer 1: The buffer is empty! Jam...
Producer 5: Produce a product ID32!
The producer 5 loads the product ID32 and the buffer location 1！
                                Consumer 4: The buffer is empty! Jam...
Producer 1...
                                Consumer 5: The buffer is empty! Jam...
Producer 3 production wake up!
Producer 3: Produce a product ID33!
The producer 3 loads the product ID33 and the buffer location 2！
^C
kafle-samrat@kaflesamrat:~/Documents/C folder$ 
```