OpenEuler

操作系统实验汇报

学号：2023211603　小组分工：王何佳100%　班级：2023211804

1 实验内容

2 实验过程

3 问题与解决

实验内容

01 完成openEuler操作系统的安装

02 完成内核更新（源代码更新方式）

03 内核模块编程、内存管理、
中断和异常处理、内核时间管理

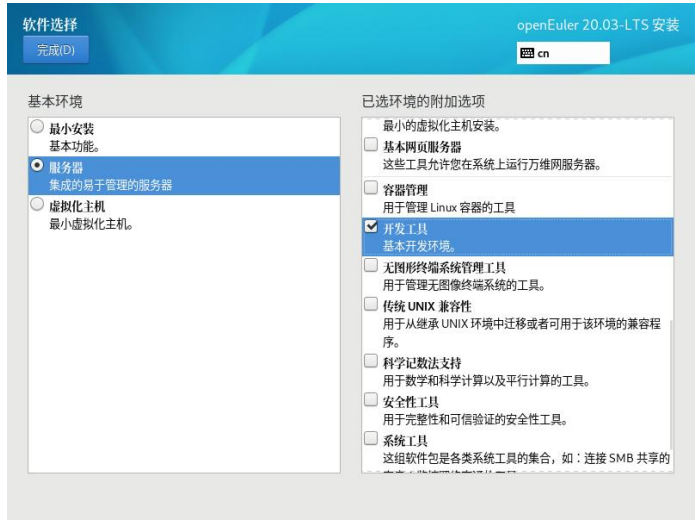
实验过程

下载镜像并创建系统

01

配置openEuler

02



安装 openEuler 操作系统

创建用户

03

执行简单命令

04

```
[bupt_wanghejia2023211603@localhost ~]$ uname -a
Linux localhost.localdomain 4.19.90-2003.4.0.0036.oe1.x86_64 #1 SMP M
_64 x86_64 x86_64 GNU/Linux
```

```
[bupt_wanghejia2023211603@localhost ~]$ getconf PAGESIZE
4096
```

实验过程

安装便捷工具，为后续实验打下基础

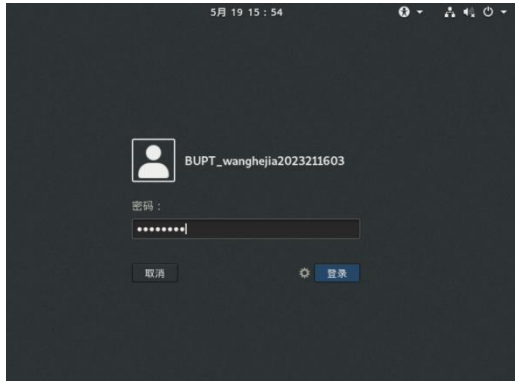
安装图形化界面

```
[osrepo]
name=osrepo
baseurl=https://mirrors.tuna.tsinghua.edu.cn/openeu
enabled=1
gpgcheck=1
gpgkey=https://mirrors.tu
Euler
```

1.配置清华源

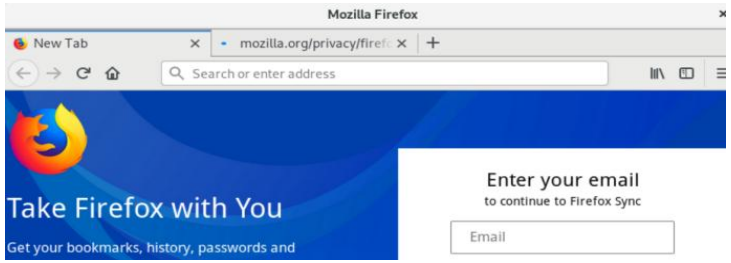
```
                                                speexdsp-1.2.0-1.oe1.x86_64
                                                switcheroo-control-1.1-7.oe1.x86_64
                                                totem-pl-parser-3.26.1-5.oe1.x86_64
                                                upower-0.99.8-5.oe1.x86_64
                                                webrtc-audio-processing-0.3.1-3.oe1.x86_64
                                                xorg-x11-drv-libinput-0.28.0-5.oe1.x86_64
                                                xorg-x11-server-1.20.6-4.oe1.x86_64
                                                xorg-x11-xauth-1:1.1-1.oe1.x86_64
                                                xorg-x11-xinit-1.4.0-5.oe1.x86_64
                                                xorg-x11-xkb-utils-7.7-28.oe1.x86_64
                                                zenity-3.30.0-2.oe1.x86_64
Installed:
  gnome-terminal-3.30.1-3.oe1.x86_64
  exiv2-0.26-17.oe1.x86_64
  gvfs-1.40.2-6.oe1.x86_64
  libcdio-2.0.0-8.oe1.x86_64
  libexif-0.6.21-20.oe1.x86_64
  libgsf-1.14.43-4.oe1.x86_64
  libiptcdata-1.0.5-1.oe1.x86_64
  nautilus-3.33.90-3.oe1.x86_64
  osinfo-db-tools-1.2.0-3.oe1.x86_64
  poppler-data-0.4.9-4.oe1.noarch
  taglib-1.11.1-12.oe1.x86_64
  tracker-miners-2.1.5-6.oe1.x86_64

Complete!
[bupt_wanghejia20232116030localhost ~]$ _
```

2.安装gnome
3.安装terminal
4.设置开机自启动
5.补全丢失文件

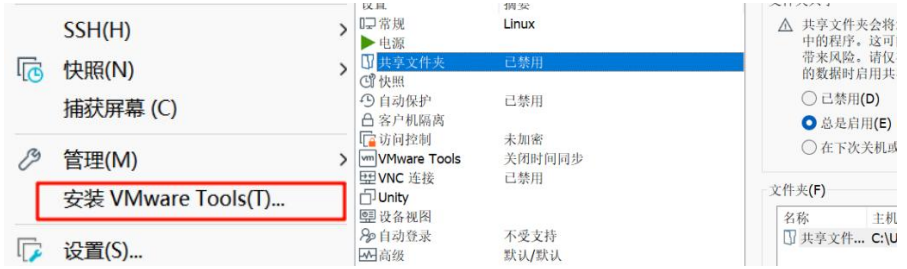


安装火狐Firefox

```
yum -y install firefox
```



安装VMware Tools

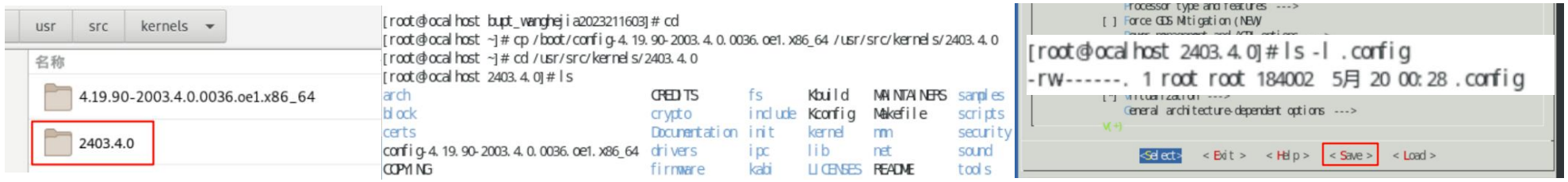


实验过程

更新内核

下载新版内核 → 重命名并移动文件 → 复制原配置文件 → 安装依赖，更新配置

准备工作



安装编译所需组件 ⟶ 编译安装 ⟶ 安装模块 ⟶ 安装内核

编译安装

```
yum install elfutils-libelf-devel  make
yum install openssl-devel
yum install bc
```

```
CC      sound/xen/snd_xen_front.mod.o
LD [M]  sound/xen/snd_xen_front.ko
CC      virt/lib/irqbypass.mod.o
LD [M]  virt/lib/irqbypass.ko
[root@ocalhost 2403.4.0]#
```

```
make modules_install
```

```
INSTALL sound/xen/snd_xen_front.ko
INSTALL virt/lib/irqbypass.ko
DEPMOD  4.19.90
```

```
make install
```

```
[root@ocalhost 2403.4.0]# make install
sh ./arch/x86/boot/install.sh 4.19.90 arch/x8
        System.map "/boot"
```

更新引导 ⟶ 重启 ⟶ 查看内核版本，安装成功

完成验证

```
openEuler (4.19.90-2003.4.0.0036.oe1.x86_64) 20.03 (LTS)
openEuler (4.19.90) 20.03 (LTS)
openEuler (4.19.90.old) 20.03 (LTS)
openEuler (0-rescue-b84a447069fd4a05
```

```
[root@ocalhost bupt_wanghejia2023211603]# grub2-editenv list
saved_entry=openEuler (4.19.90) 20.03 (LTS)
boot_success=0
```

```
[root@ocalhost bupt_wanghejia2023211603]# uname -a
Linux localhost.localdomain 4.19.90 #1 SMP Tue May 20 00:38:07 CST 2025 x86_64 x86_64 x86_64 GNU/Linux
```

实验过程

基础实验

01

内核模块编程

02

内存管理

03

中断和异常处理

04

内核时间管理

实验过程

内核模块编程

helloworld.c

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

MODULE_LICENSE("GPL");

static int __init hello_init(void){
    printk(KERN_INFO "hello init\n");
    printk(KERN_INFO "hello, world!\n");
    return 0;
}

static void __exit hello_exit(void){
    printk(KERN_INFO "hello exit\n");
}
module_init(hello_init);
module_exit(hello_exit);
```

Makefile

```
ifneq ($(KERNELRELEASE),)
    obj-m := helloworld.o
else
    KERNELDIR ?=/usr/src/kernels/2403.4.0
    PWD := $(shell pwd)
default:
    $(MAKE) -C $(KERNELDIR) M=$(PWD) modules
endif
.PHONY:clean
clean:
    -rm *.mod.c *.o *.order *.symvers *.ko
```

编译 make

```
[bupt_wanghejia2023211603@ocalhost ~]$ sudo make
[sudo] bupt_wanghejia2023211603 的密码：
make -C /usr/src/kernels/2403.4.0 M=/home/bupt_wanghejia2023211603 modules
make[1]: 进入目录"/usr/src/kernels/2403.4.0"
  CC [M]  /home/bupt_wanghejia2023211603/helloworld.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/bupt_wanghejia2023211603/helloworld.mod.o
[bupt_wanghejia2023211603@ocalhost ~]$ ll
总用量 452K
-rwxr-xr-x. 1 bupt_wanghejia2023211603 bupt_wanghejia2023211603  309 5月 20 14:25 '#helloworld.c#'
-rwxrwxrwx. 1 bupt_wanghejia2023211603 bupt_wanghejia2023211603  331 5月 20 14:21  helloworld.c
-rw-------. 1 root                     root                     210K 5月 20 14:22  helloworld.ko
-rw-------. 1 root                     root                      843 5月 20 14:22  helloworld.mod.c
-rw-------. 1 root                     root                     106K 5月 20 14:22  helloworld.mod.o
-rw-------. 1 root                     root                     106K 5月 20 14:22  helloworld.o
-rwxrwxrwx. 1 bupt_wanghejia2023211603 bupt_wanghejia2023211603  251 5月 20 14:09  Makefile
-rwxrwxrwx. 1 bupt_wanghejia2023211603 bupt_wanghejia2023211603  254 5月 20 14:09  Makefile~
```

加载内核模块 `insmod helloworld.ko`

查看打印信息 `dmesg | tail -n 2`

查看内核模块 `lsmod | tail -n 2`

卸载内核模块 `rmmod helloworld`

```
[root@ocalhost bupt_wanghejia2023211603]# insmod helloworld.ko
[root@ocalhost bupt_wanghejia2023211603]# dmesg | tail -n 2
[ 3144.824612] hello init
[ 3144.824613] hello, world!
[root@ocalhost bupt_wanghejia2023211603]# lsmod | tail -n 2
dm_log                 20480  2 dm_region_hash,dm_mirror
dm_mod                155648  8 dm_log,dm_mirror
[root@ocalhost bupt_wanghejia2023211603]# rmmod helloworld
```

实验过程

内存管理

kmalloc.c 分配1KB,8KB

```c
#include <linux/module.h>
#include <linux/slab.h>
MODULE_LICENSE("GPL");
unsigned char *kmallocmem1;
unsigned char *kmallocmem2;
static int __init mem_module_init(void){
    printk(KERN_INFO "Start kmalloc!\n");
    kmallocmem1 = (unsigned char*)kmalloc(1024, GFP_KERNEL);
    if (kmallocmem1 != NULL) {
        printk(KERN_ALERT "kmallocmem1 addr = %p\n", (void *)kmallocmem1);
    } else {
        printk(KERN_ERR "Failed to allocate kmallocmem1!\n");
    }
    kmallocmem2 = (unsigned char *)kmalloc(8192, GFP_KERNEL);
    if (kmallocmem2 != NULL) {
        printk(KERN_ALERT "kmallocmem2 addr = %p\n", (void *)kmallocmem2);
    } else {
        printk(KERN_ERR "Failed to allocate kmallocmem2!\n");
    }
    return 0;
}
static void __exit mem_module_exit(void){
    if (kmallocmem1) {
        kfree(kmallocmem1);
    }
    if (kmallocmem2) {
        kfree(kmallocmem2);
    }
    printk(KERN_INFO "Exit kmalloc!\n");
}
module_init(mem_module_init);
module_exit(mem_module_exit);
```

编译运行

```
make
insmod kmalloc.ko
dmesg | tail -n 3
rmmod kmalloc
dmesg | tail -n 4
```

查看内存布局

```
Virtual memory map with 4 level page tables:

0000000000000000 - 00007fffffffffff (=47 bits) user space, different per
hole caused by [47:63] sign extension
ffff800000000000 - ffff87ffffffffff (=43 bits) guard hole, reserved for
ffff880000000000 - ffff887fffffffff (=39 bits) LDT remap for PTI
ffff888000000000 - ffffc87fffffffff (=64 TB) direct mapping of all phys.
ffffc88000000000 - ffffc8ffffffffff (=39 bits) hole
ffffc90000000000 - ffffe8ffffffffff (=45 bits) vmalloc/ioremap space
ffffe90000000000 - ffffe9ffffffffff (=40 bits) hole
ffffea0000000000 - ffffeaffffffffff (=40 bits) virtual memory map (1TB)
... unused hole ...
ffffec0000000000 - fffffbffffffffff (=44 bits) kasan shadow memory (16TB
... unused hole ...
```

分配的内存地址位于内核空间

vmalloc.c 分配8KB,1MB,64MB

```c
#include <linux/module.h>
#include <linux/vmalloc.h>
MODULE_LICENSE("GPL");
unsigned char *vmallocmem1;
unsigned char *vmallocmem2;
unsigned char *vmallocmem3;
static int __init mem_module_init(void){
    printk(KERN_INFO "Start vmalloc!\n");
    vmallocmem1 = (unsigned char*)vmalloc(8192);
    if (vmallocmem1 != NULL) {
        printk(KERN_INFO "vmallocmem1 addr = %lx\n", (unsigned long)vmallocmem1);
    } else {
        printk(KERN_ERR "Failed to allocate vmallocmem1!\n");
    }
    vmallocmem2 = (unsigned char*)vmalloc(1048576);
    if (vmallocmem2 != NULL) {
        printk(KERN_INFO "vmallocmem2 addr = %lx\n", (unsigned long)vmallocmem2);
    } else {
        printk(KERN_ERR "Failed to allocate vmallocmem2!\n");
    }
    vmallocmem3 = (unsigned char*)vmalloc(67108864);
    if (vmallocmem3 != NULL) {
        printk(KERN_INFO "vmallocmem3 addr = %lx\n", (unsigned long)vmallocmem3);
    } else {
        printk(KERN_ERR "Failed to allocate vmallocmem3!\n");
    }
    return 0;
}
static void __exit mem_module_exit(void){
    vfree(vmallocmem1);
    vfree(vmallocmem2);
    vfree(vmallocmem3);
    printk(KERN_INFO "Exit vmalloc!\n");
}
module_init(mem_module_init);
module_exit(mem_module_exit);
```

编译运行

```
make
insmod vmalloc.ko
dmesg | tail -n 4
rmmod vmalloc
dmesg | tail -n 5
```

```
[root@localhost test1]# insmod | tail -n 4
insmod: ERROR: missing filename.
[root@localhost test1]# insmod vmalloc.ko
[root@localhost test1]# dmesg | tail -n 4
[ 7216.464455] Start vmalloc!
[ 7216.464461] vmallocmem1 addr = ffffb84140679000
[ 7216.464487] vmallocmem2 addr = ffffb84142a69000
[ 7216.465653] vmallocmem3 addr = ffffb84150001000
[root@localhost test1]# rmmod vmalloc
[root@localhost test1]# dmesg | tail -n 5
[ 7216.464455] Start vmalloc!
```

查看系统页表大小

```
[root@localhost test1]# getconf PAGE_SIZE
4096
```

分配的内存地址位于内核空间

实验过程

中断和异常处理

用tasklet打印helloworld

```c
#include <linux/module.h>
#include <linux/interrupt.h>
MODULE_LICENSE("GPL");
static struct tasklet_struct my_tasklet;
static void tasklet_handler(unsigned long data){
    printk(KERN_INFO "Hello World! tasklet is working...\n");
}
static int __init mytasklet_init(void){
    printk(KERN_INFO "Start tasklet module...\n");
    tasklet_init(&my_tasklet, tasklet_handler, 0);
    tasklet_schedule(&my_tasklet);
    return 0;
}
static void __exit mytasklet_exit(void){
    tasklet_kill(&my_tasklet);
    printk(KERN_INFO "Exit tasklet module...\n");

}
module_init(mytasklet_init);
module_exit(mytasklet_exit);
```

```
[root@localhost test3]# make
make -C /usr/src/kernels/2403.4.0 M=/home/bupt_wanghejia2023211603/test3 modules
make[1]: 进入目录"/usr/src/kernels/2403.4.0"
  CC [M]  /home/bupt_wanghejia2023211603/test3/tasklet_intertupt.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/bupt_wanghejia2023211603/test3/tasklet_intertupt.mod.o
  LD [M]  /home/bupt_wanghejia2023211603/test3/tasklet_intertupt.ko
make[1]: 离开目录"/usr/src/kernels/2403.4.0"
[root@localhost test3]# insmod tasklet_intertupt.ko
[root@localhost test3]# dmesg | tail -n 2
[ 7858.368072] Start tasklet module...
[ 7858.368092] Hello World! tasklet is working..
[root@localhost test3]# rmmod tasklet_intertupt
[root@localhost test3]# dmesg | tail -n 3
[ 7858.368072] Start tasklet module...
[ 7858.368092] Hello World! tasklet is working..
[ 7878.687021] Exit tasklet module...
```

工作队列周期打印helloworld

```c
#include <linux/module.h>
#include <linux/workqueue.h>
#include <linux/delay.h>
MODULE_LICENSE("GPL");
static struct workqueue_struct *queue = NULL;
static struct delayed_work mywork;
static int i = 0;
// work handle
void work_handle(struct work_struct *work){
    printk(KERN_ALERT "Hello World!\n");
}
static int __init timewq_init(void){
    printk(KERN_ALERT "Start workqueue_test module.\n");
    queue = create_singlethread_workqueue("workqueue_test");
    if(queue == NULL){
        printk(KERN_ALERT "Failed to create workqueue_test!\n");
        return -1;
    }
    INIT_DELAYED_WORK(&mywork, work_handle);
    for(;i <= 3; i++){
        queue_delayed_work(queue, &mywork, 5 * HZ);
        ssleep(15);
    }
    return 0;
}
static void __exit timewq_exit(void){
    flush_workqueue(queue);
    destroy_workqueue(queue);
    printk(KERN_ALERT "Exit workqueue_test module.\n");
}
module_init(timewq_init);
module_exit(timewq_exit);
```

```
[root@localhost test]# dmesg
[ 9342.994893] Start workque
[ 9348.049683] Hello World!
[ 9363.409676] Hello World!
[ 9378.763324] Hello World!
[ 9394.121182] Hello World!
```

捕获终端按键信号

```c
#include <signal.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
void signal_handler(int sig){
    switch(sig) {
        case SIGINT:
            printf("\nGet a signal:SIGINT. You pressed ctrl+c.\n");
            break;
        case SIGQUIT:
            printf("\nGet a signal:SIGQUIT. You pressed ctrl+\\.\n");
            break;
        case SIGTSTP:
            printf("\nGet a signal:SIGTSTP. You pressed ctrl+z.\n");
            break;
    }
    exit(0);
}
int main(){
    printf("Current process ID is %d\n", getpid());
    signal(SIGINT, signal_handler);
    signal(SIGQUIT, signal_handler);
    signal(SIGTSTP, signal_handler);
    for(;;);
    return 0;
}
```

```
总用量 36K
-rwx------. 1 root                    root                    17K 5月  20 16:29 catch_signal
-rwxrwxrwx. 1 bupt_wanghejia2023211603 bupt_wanghejia2023211603 791 5月  20 16:27 catch_signal.c
-rwxrwxrwx. 1 bupt_wanghejia2023211603 bupt_wanghejia2023211603 253 5月  20 16:28 Makefile
-rwxrwxrwx. 1 bupt_wanghejia2023211603 bupt_wanghejia2023211603 251 5月  20 14:09 Makefile-
-rw-------. 1 root                    root                     59 5月  20 16:28 modules.order
[root@localhost test]#./catch_signal
Current process ID is 28024
^C
Get a signal:SIGINT. You pressed ctrl+c.
[root@localhost test]#./catch_signal
Current process ID is 28037
^C
Get a signal:SIGINT. You pressed ctrl+c.
[root@localhost test]#./catch_signal
Current process ID is 28050
^C
Get a signal:SIGINT. You pressed ctrl+c.
[root@localhost test]#jobs
```

实验过程

内核时间管理

调用时钟接口打印当前时间

```c
#include <linux/module.h>
#include <linux/time.h>
#include <linux/rtc.h>
MODULE_LICENSE("GPL");
struct timeval tv;
struct rtc_time tm;
static int __init currenttime_init(void){
    int year, mon, day, hour, min, sec;
    printk(KERN_INFO "Start current_time module...\n");
    do_gettimeofday(&tv);
    rtc_time_to_tm(tv.tv_sec, &tm);
    year = tm.tm_year + 1900;
    mon = tm.tm_mon + 1;
    day = tm.tm_mday;
    hour = tm.tm_hour + 8;
    min = tm.tm_min;
    sec = tm.tm_sec;
    printk(KERN_INFO "Current time: %d-%02d-%02d %02d:%02d:%02d\n"
            , year, mon, day, hour, min, sec);
    return 0;
}
static void __exit currenttime_exit(void){
    printk(KERN_INFO "Exit current_time module...\n");
}
module_init(currenttime_init);
module_exit(currenttime_exit);
```

```
  LD [M]  /home/bupt_wanghejia2023211603/test/current_time.ko
make[1]: 离开目录"/usr/src/kernels/2403.4.0"
[root@localhost test]# insmod current_time.ko
[root@localhost test]# dmesg | tail -n 2
[10417.678233] Start current_time module...
[10417.678236] Current time: 2025-05-20 16:39:45
[root@localhost test]# rmmod current_time
[root@localhost test]# dmesg | tail -n 3
[10417.678233] Start current_time module...
[10417.678236] Current time: 2025-05-20 16:39:45
[10433.776873] Exit current_time module...
```

在特定时刻打印helloworld

```c
#include <linux/module.h>
#include <linux/timer.h>
MODULE_LICENSE("GPL");
struct timer_list timer;
void print(struct timer_list *timer){
    printk(KERN_INFO "hello, world!\n");
}
static int __init timer_init(void){
    printk(KERN_INFO "Start timer_example module...\n");
    timer.expires = jiffies + 10 * HZ;
    timer.function = print;
    add_timer(&timer);
    return 0;
}
static void __exit timer_exit(void){
    printk(KERN_INFO "Exit timer_example module...\n");
}
module_init(timer_init);
module_exit(timer_exit);
```

```
[root@localhost test]# make
make -C /usr/src/kernels/2403.4.0 M=/home/bupt_wanghejia2023211603/test modules
make[1]: 进入目录"/usr/src/kernels/2403.4.0"
  CC [M]  /home/bupt_wanghejia2023211603/test/timer_example.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/bupt_wanghejia2023211603/test/timer_example.mod.o
  LD [M]  /home/bupt_wanghejia2023211603/test/timer_example.ko
make[1]: 离开目录"/usr/src/kernels/2403.4.0"
[root@localhost test]# insmod timer_example.ko
[root@localhost test]# dmesg -t | tail -n 2
perf: interrupt took too long (3229 > 3126), lowering kernel.perf_event_max_sample_rate to 61000
Start timer_example module...
[root@localhost test]# dmesg -T | tail -n 2
[二 5月 20 16:45:53 2025] Start timer_example module...
[二 5月 20 16:46:04 2025] hello, world!
[root@localhost test]# rmmod timer_example
[root@localhost test]# dmesg -T | tail -n 3
[二 5月 20 16:45:53 2025] Start timer_example module...
[二 5月 20 16:46:04 2025] hello, world!
[二 5月 20 16:46:41 2025] Exit timer_example module...
```

10秒后打印 “hello,world!”

监控累加计算代码的运行时间

```c
#include <linux/module.h>
#include <linux/time.h>
MODULE_LICENSE("GPL");
#define NUM 100000
struct timeval tv;
static long sum(int num){
    int i;
    long total = 0;
    for (i = 1; i <= num; i++)
        total = total + i;
    printk(KERN_INFO "The sum of 1 to %d is: %ld\n", num, total);
    return total;
}
static int __init sum_init(void){
    int start;
    int start_u;
    int end;
    int end_u;
    long time_cost;
    long s;
    printk(KERN_INFO "Start sum_time module...\n");
    do_gettimeofday(&tv);
    start = (int)tv.tv_sec;
    start_u = (int)tv.tv_usec;
    printk(KERN_INFO "The start time is: %d s %d us\n", start, start_u);
    s = sum(NUM);
    do_gettimeofday(&tv);
    end = (int)tv.tv_sec;
    end_u = (int)tv.tv_usec;
    printk(KERN_INFO "The end time is: %d s %d us\n", end, end_u);
    time_cost = (end - start) * 1000000 + (end_u - start_u);
    printk(KERN_INFO "The cost time of sum from 1 to %d is: %ld us\n"
            , NUM, time_cost);
    return 0;
}
static void __exit sum_exit(void){
    printk(KERN_INFO "Exit sum_time module...\n");
}
module_init(sum_init);
module_exit(sum_exit);
```

```
[11119.412793] The sum of 1 to 100000 is: 5000050000
[11119.412794] The end time is: 1747731087 s 147000 us
[11119.412794] The cost time of sum from 1 to 100000 is: 1 us
[11131.308276] Exit sum_time module...
```

问题与解决

权限问题

无法更改文件内容

```
[osrepo]
name=osrepo
baseurl=https://mirrors.tuna.tsinghua.edu.cn/openeuler/openEuler-20.03-LTS/OS/x86_64/
enabled=1
gpgcheck=1
gpgkey=https://mirrors/tuna.tsinghua.edu.cn/openeuler/openEuler-20.03-LTS/OS/x86_64/RPM-GPG-KEY-open
Euler
```

```
E45: 'readonly' option is set (add ! to override)
```

:wq!强制保存，又出现报错

```
"/etc/yum.repos.d/openEuler_x86_64.repo"
"/etc/yum.repos.d/openEuler_x86_64.repo" E212: Can't open file for writing
Press ENTER or type command to continue_
```

原因：权限不够

解决：先:q!强制退出，再启用sudo权限

```
sudo vim /etc/yum.repos.d/openEuler_x86_64.repo
```

```
[bupt_wanghejia2023211603@localhost ~]$ sudo vim /etc/yum.repos.d/openEuler_x86_64.repo

We trust you have received the usual lecture from the local System
Administrator. It usually boils down to these three things:

    #1) Respect the privacy of others.
    #2) Think before you type.
    #3) With great power comes great responsibility.
```

对文件操作时出错

移动文件夹时报错“No such file or directory”

原因：下载时未开启sudo权限

解决：重新用sudo权限下载并执行mv

提示overwrite输入y确认覆盖

```
[root@localhost tmp]# wget https://gitee.com/name1e5s/xsession/raw/master/Xsession
--2025-05-19 15:42:41--  https://gitee.com/name1e5s/xsession/raw/master/Xsession
Resolving gitee.com (gitee.com)... 180.76.199.13, 180.76.198.77, 180.76.198.225
Connecting to gitee.com (gitee.com)|180.76.199.13|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/plain]
Saving to: ‘Xsession’

Xsession                    [ <=>                                ]   5.02K  --.-KB/s    in 0s

2025-05-19 15:42:42 (62.5 MB/s) - ‘Xsession’ saved [5145]

[root@localhost tmp]# mv Xsession /etc/gdm/Xsession
mv: overwrite '/etc/gdm/Xsession'? y
```

其他权限问题

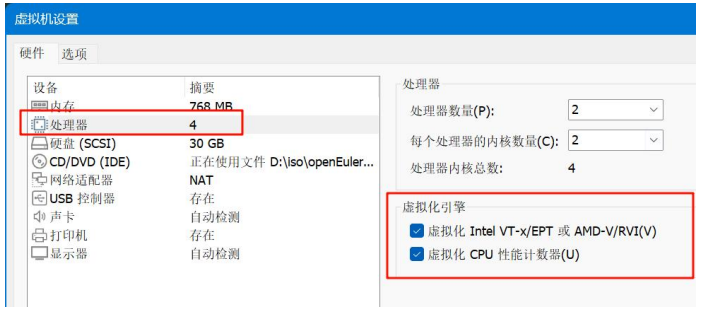
问题：多数操作需要root权限　**解决**：sudo su后进行操作

问题与解决

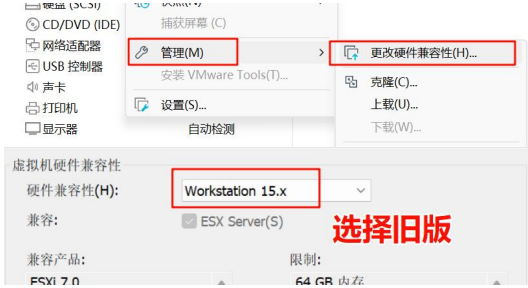
禁用CPU

为解决这个问题进行了**诸多**尝试

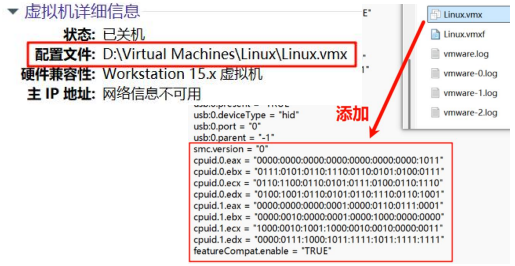
1.打开虚拟化CPU性能计数器



2.更改硬件兼容性



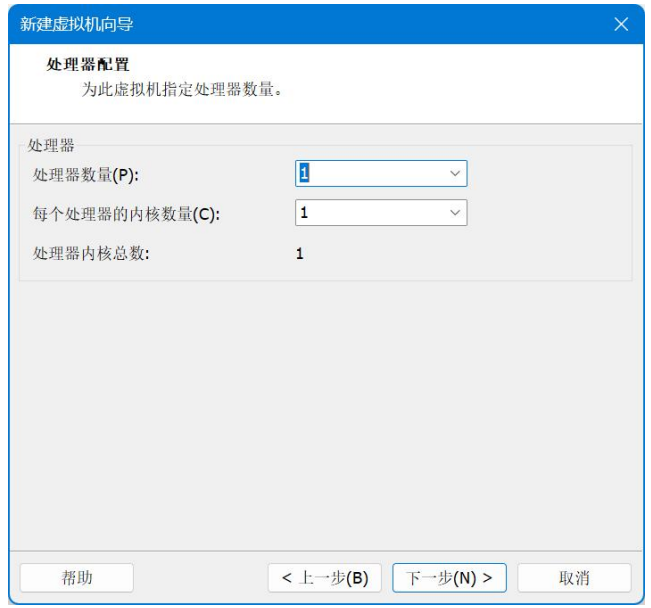
3.更改配置文件



以上常规方法**都没能**解决问题

最终解决方案

重装系统
分配更少的CPU以保证稳定性

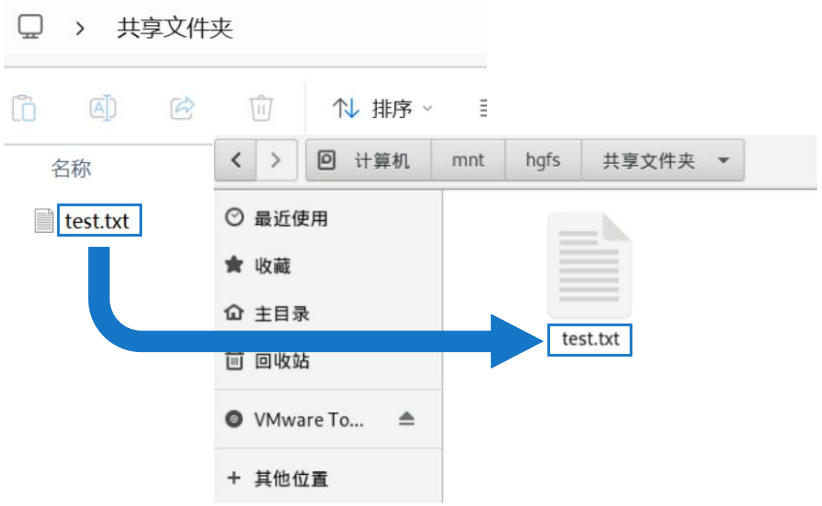


问题与解决

代码报错

虚拟机界面编程不便

安装VMwareTools，使用共享文件夹
在本机编程完同步到虚拟机



同步之后仍有很多报错

根据提示进行修改即可

```
[bupt_wanghejia2023211603@localhost ~]$ make
Makefile:7: *** 遗漏分隔符（null）。 停止。

[bupt_wanghejia2023211603@localhost ~]$ sudo make
[sudo] bupt_wanghejia2023211603 的密码：
make -C /usr/src/kernels/2403.4.0 M=/home/bupt_wanghejia2023211603 modules
make[1]: 进入目录"/usr/src/kernels/2403.4.0"
make[2]: *** 没有规则可制作目标"/home/bupt_wanghejia2023211603/helloworld.c"，由
'/home/bupt_wanghejia2023211603/helloworld.o' 需求。 停止。
make[1]: *** [Makefile:1529：_module_/home/bupt_wanghejia2023211603] 错误 2
make[1]: 离开目录"/usr/src/kernels/2403.4.0"
make: *** [Makefile:7：default] 错误 2
```

OpenEuler