《openEuler 实验》

一、实验分工

姓名：项 枫 班级：2022211801 学号：2022211570 分工：100%

二、实验环境

虚拟机 Vmware 和 openEuler 操作系统。

三、实验过程

（一）openEuler 操作系统安装与内核编译

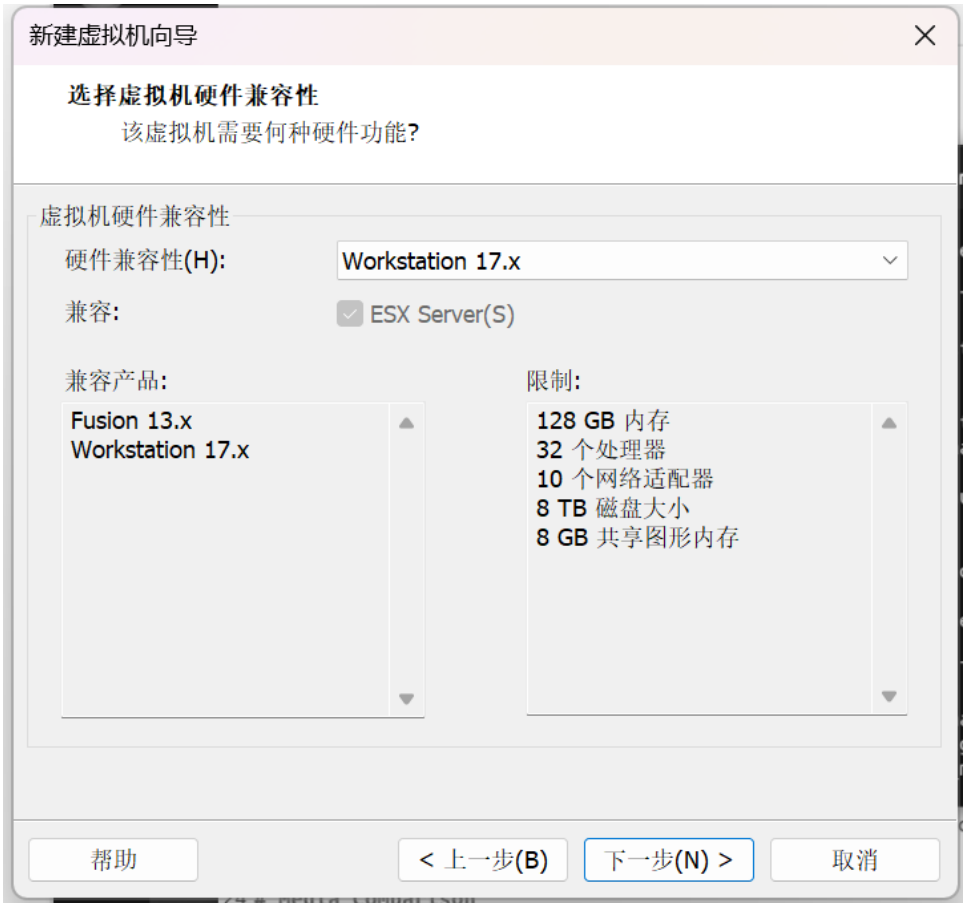
1、操作系统安装

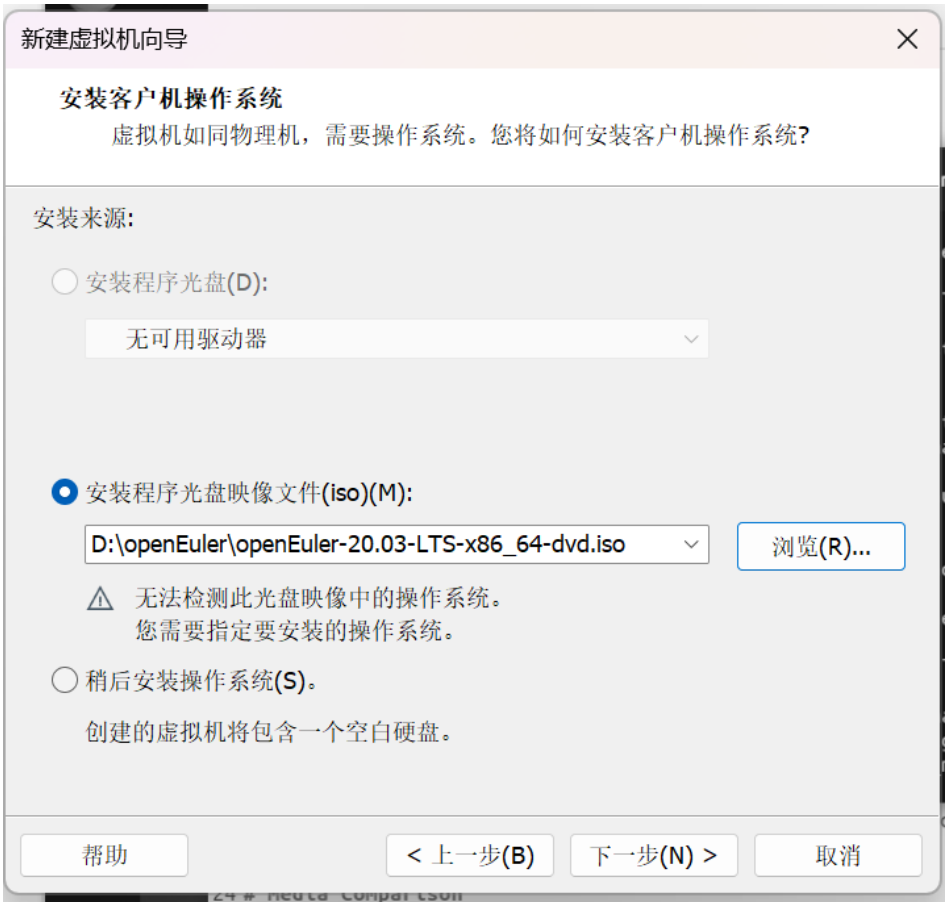
（1）下载 OpenEular 镜像

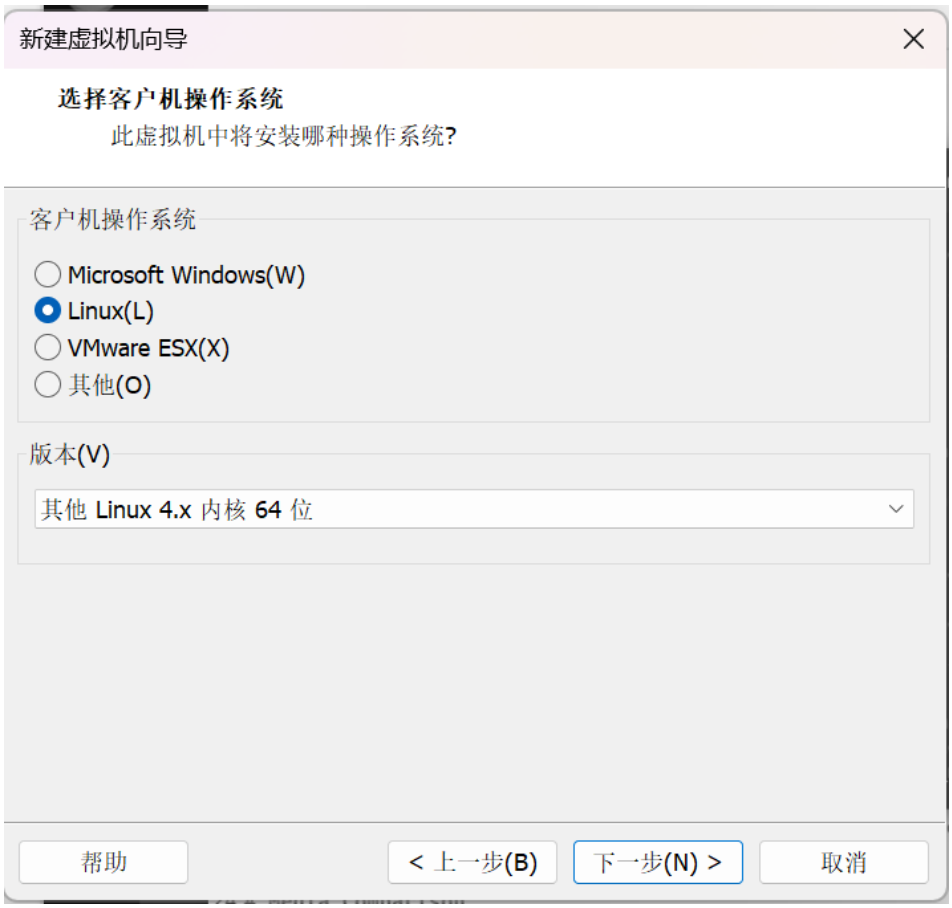
（2）安装到虚拟机

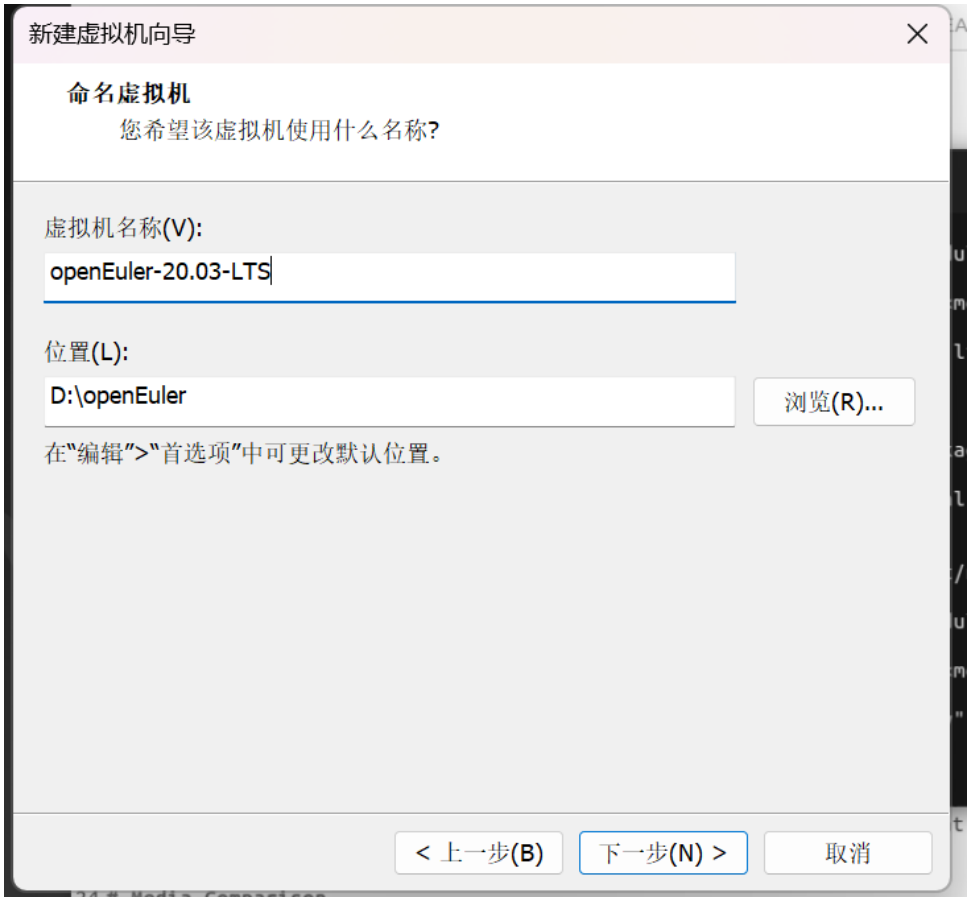
①新建虚拟机

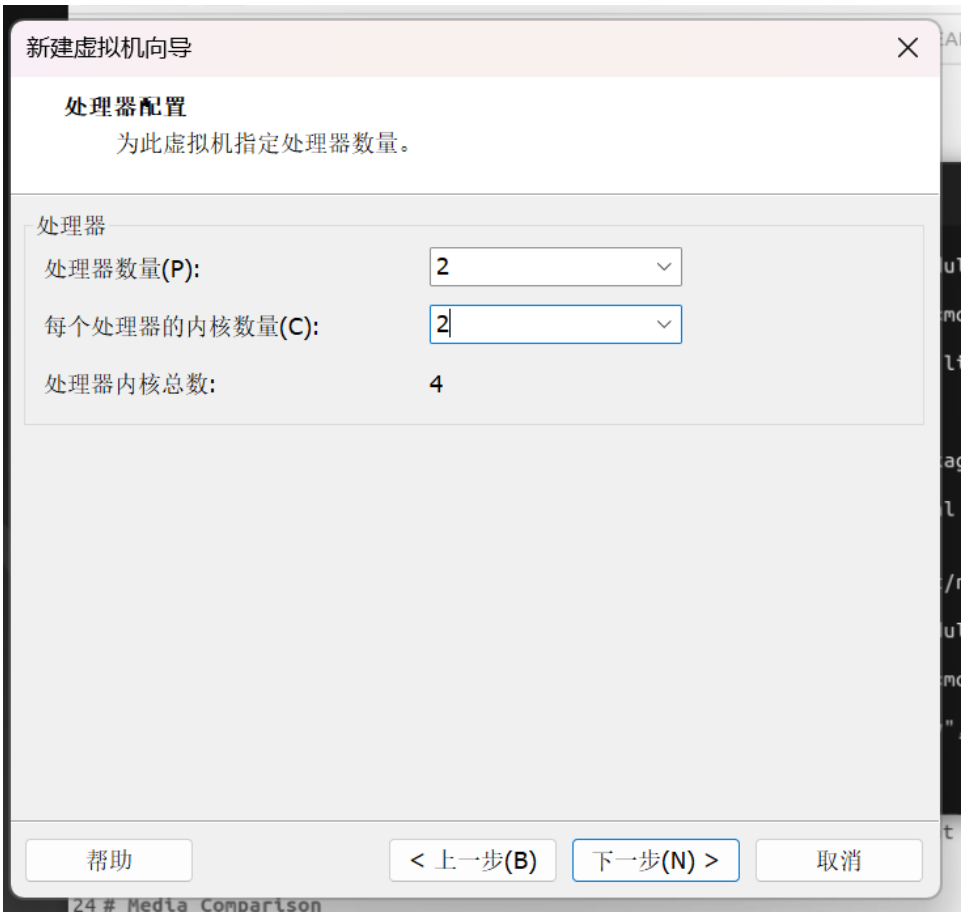

新建虚拟机向导
vmware
WORKSTATION
PRO™
17
欢迎使用新建虚拟机向导
您希望使用什么类型的配置?
典型(推荐)(T)
通过几个简单的步骤创建 Workstation 17.x 虚拟机。
自定义(高级)(C)
创建带有 SCSI 控制器类型、虚拟磁盘类型以及与旧版 VMware 产品兼容性等高级选项的虚拟机。
帮助
< 上一步(B)
下一步(N) >
取消


新建虚拟机向导
选择虚拟机硬件兼容性
该虚拟机需要何种硬件功能?
虚拟机硬件兼容性
硬件兼容性(H):
Workstation 17.x
兼容:
ESX Server(S)
兼容产品:
Fusion 13.x
Workstation 17.x
限制:
128 GB 内存
32 个处理器
10 个网络适配器
8 TB 磁盘大小
8 GB 共享图形内存
帮助
< 上一步(B)
下一步(N) >
取消


新建虚拟机向导
安装客户机操作系统
虚拟机如同物理机，需要操作系统。您将如何安装客户机操作系统?
安装来源:
安装程序光盘(D):
无可用驱动器
安装程序光盘映像文件(iso)(M):
D:\openEuler\openEuler-20.03-LTS-x86_64-dvd.iso
浏览(R)...
无法检测此光盘映像中的操作系统。
您需要指定要安装的操作系统。
稍后安装操作系统(S)。
创建的虚拟机将包含一个空白硬盘。
帮助
< 上一步(B)
下一步(N) >
取消


新建虚拟机向导
选择客户机操作系统
此虚拟机中将安装哪种操作系统?
客户机操作系统
Microsoft Windows(W)
Linux(L)
VMware ESX(X)
其他(O)
版本(V)
其他 Linux 4.x 内核 64 位
帮助
< 上一步(B)
下一步(N) >
取消

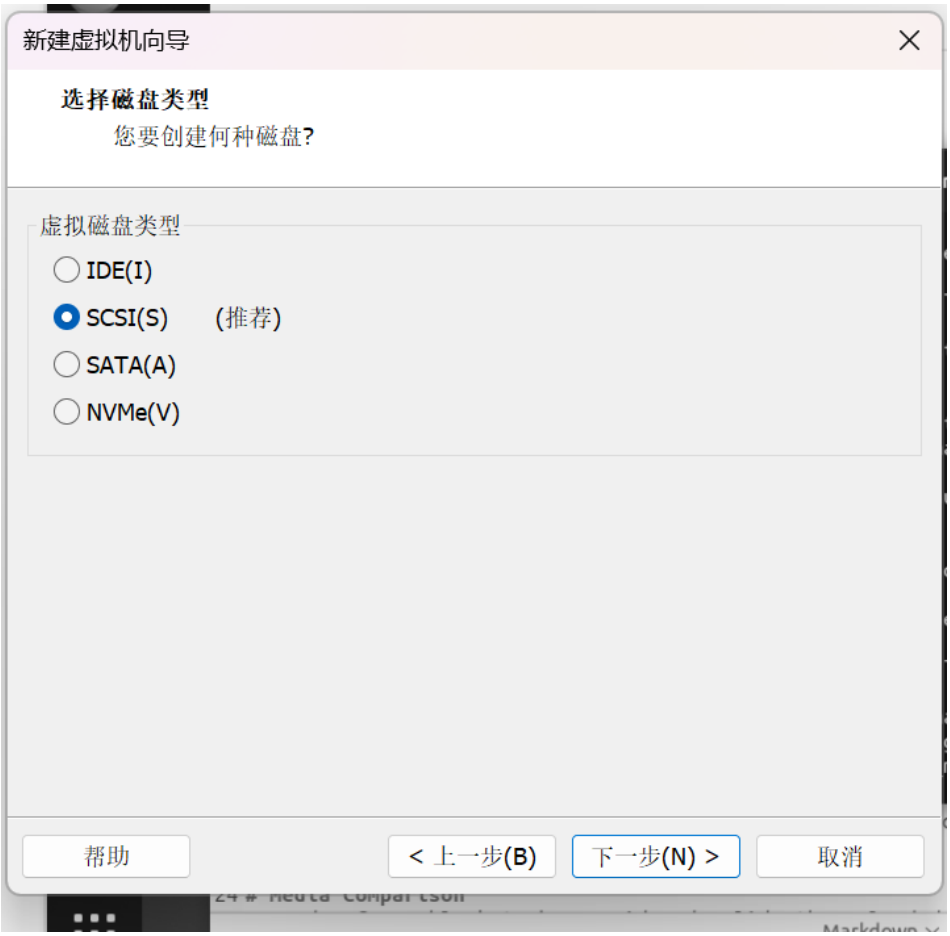

新建虚拟机向导
命名虚拟机
您希望该虚拟机使用什么名称?
虚拟机名称(V):
openEuler-20.03-LTS
位置(L):
D:\openEuler
浏览(R)...
在"编辑">"首选项"中可更改默认位置。
< 上一步(B)
下一步(N) >
取消

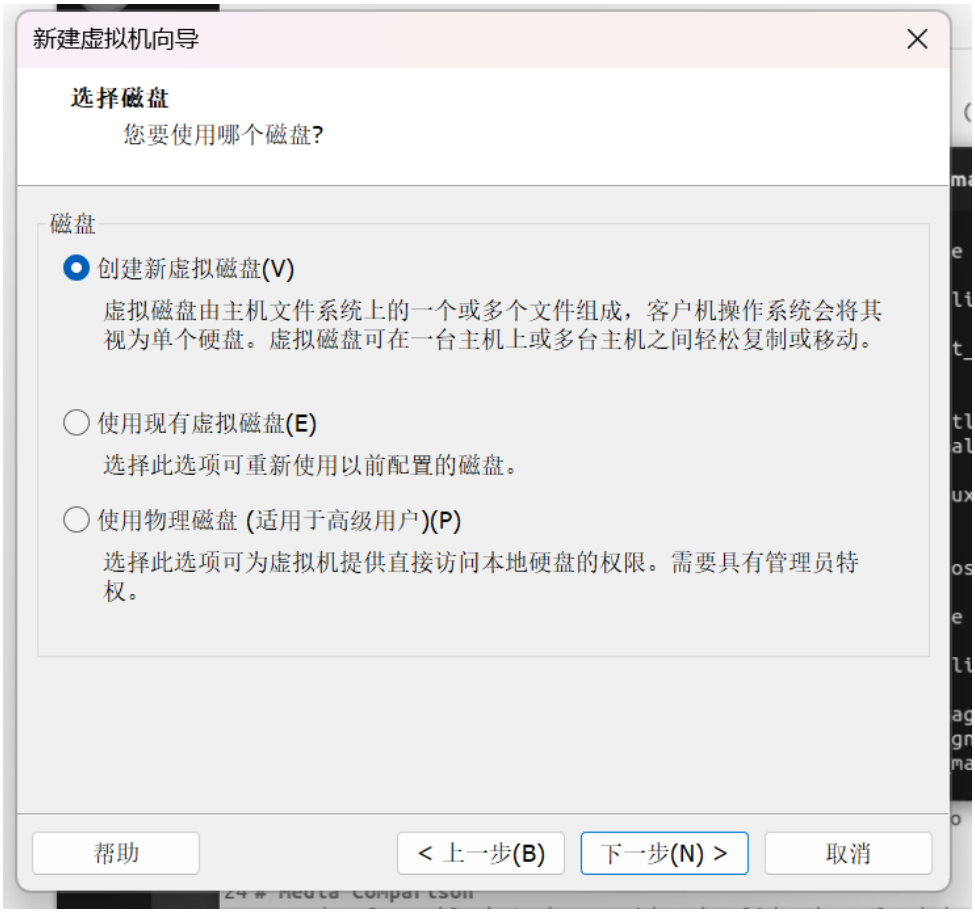

新建虚拟机向导
处理器配置
为此虚拟机指定处理器数量。
处理器
处理器数量(P):
2
每个处理器的内核数量(C):
2
处理器内核总数:
4
帮助
< 上一步(B)
下一步(N) >
取消


新建虚拟机向导
此虚拟机的内存
您要为此虚拟机使用多少内存?
指定分配给此虚拟机的内存量。内存大小必须为 4 MB 的倍数。
128 GB
64 GB
32 GB
16 GB
8 GB
4 GB
2 GB
1 GB
512 MB
256 MB
128 MB
64 MB
32 MB
16 MB
8 MB
4 MB
此虚拟机的内存(M):
4096
MB
最大推荐内存:
13.3 GB
推荐内存:
768 MB
客户机操作系统最低推荐内存:
32 MB
帮助
< 上一步(B)
下一步(N) >
取消

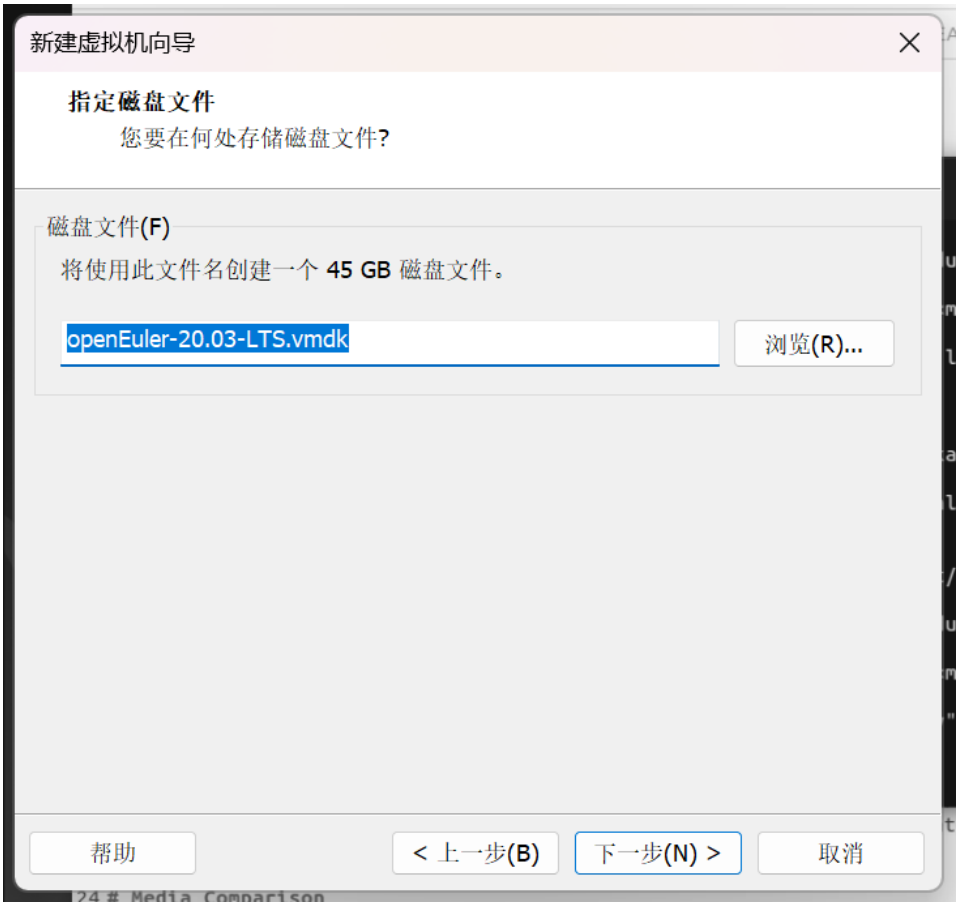

新建虚拟机向导
网络类型
要添加哪类网络?
网络连接
使用桥接网络(R)
为客户机操作系统提供直接访问外部以太网网络的权限。客户机在外部网络上必须有自己的 IP 地址。
使用网络地址转换(NAT)(E)
为客户机操作系统提供使用主机 IP 地址访问主机拨号连接或外部以太网网络连接的权限。
使用仅主机模式网络(H)
将客户机操作系统连接到主机上的专用虚拟网络。
不使用网络连接(T)
帮助
< 上一步(B)
下一步(N) >
取消

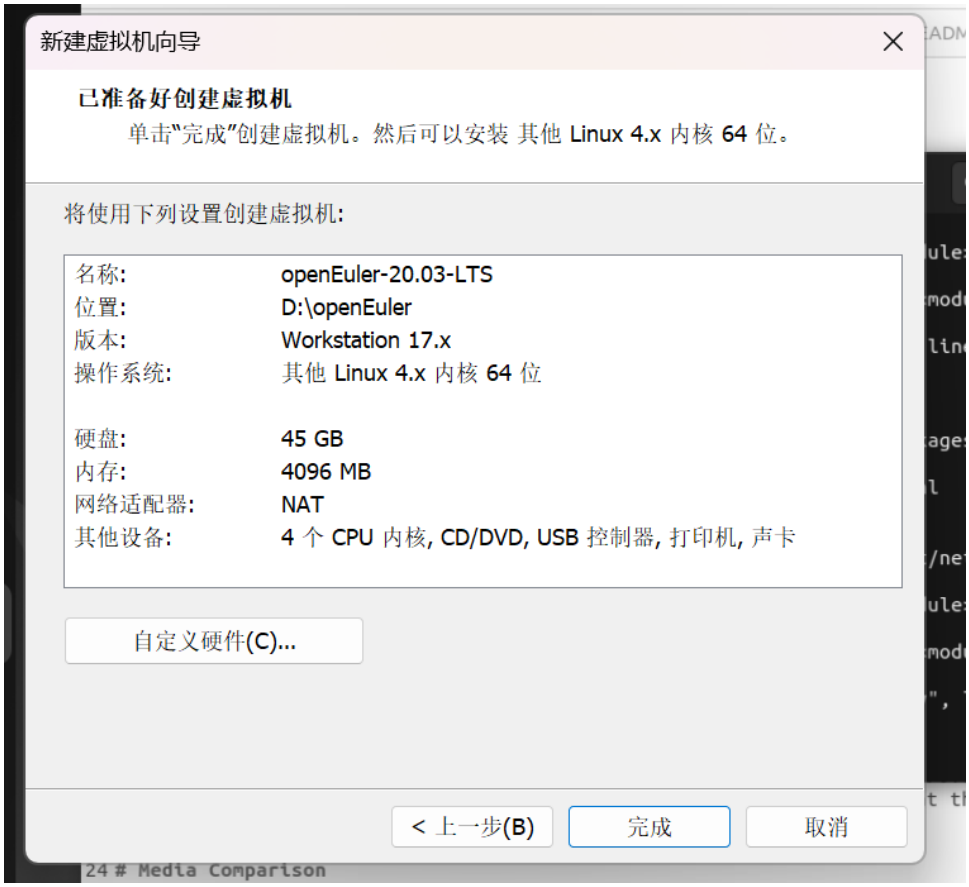

新建虚拟机向导
选择 I/O 控制器类型
您希望将哪种 SCSI 控制器类型用于 SCSI 虚拟磁盘?
I/O 控制器类型
SCSI 控制器:
BusLogic(U) (不适用于 64 位客户机)
LSI Logic(L) (推荐)
LSI Logic SAS(S)
准虚拟化 SCSI(P)
帮助
< 上一步(B)
下一步(N) >
取消


新建虚拟机向导
选择磁盘类型
您要创建何种磁盘?
虚拟磁盘类型
IDE(I)
SCSI(S) (推荐)
SATA(A)
NVMe(V)
帮助
< 上一步(B)
下一步(N) >
取消

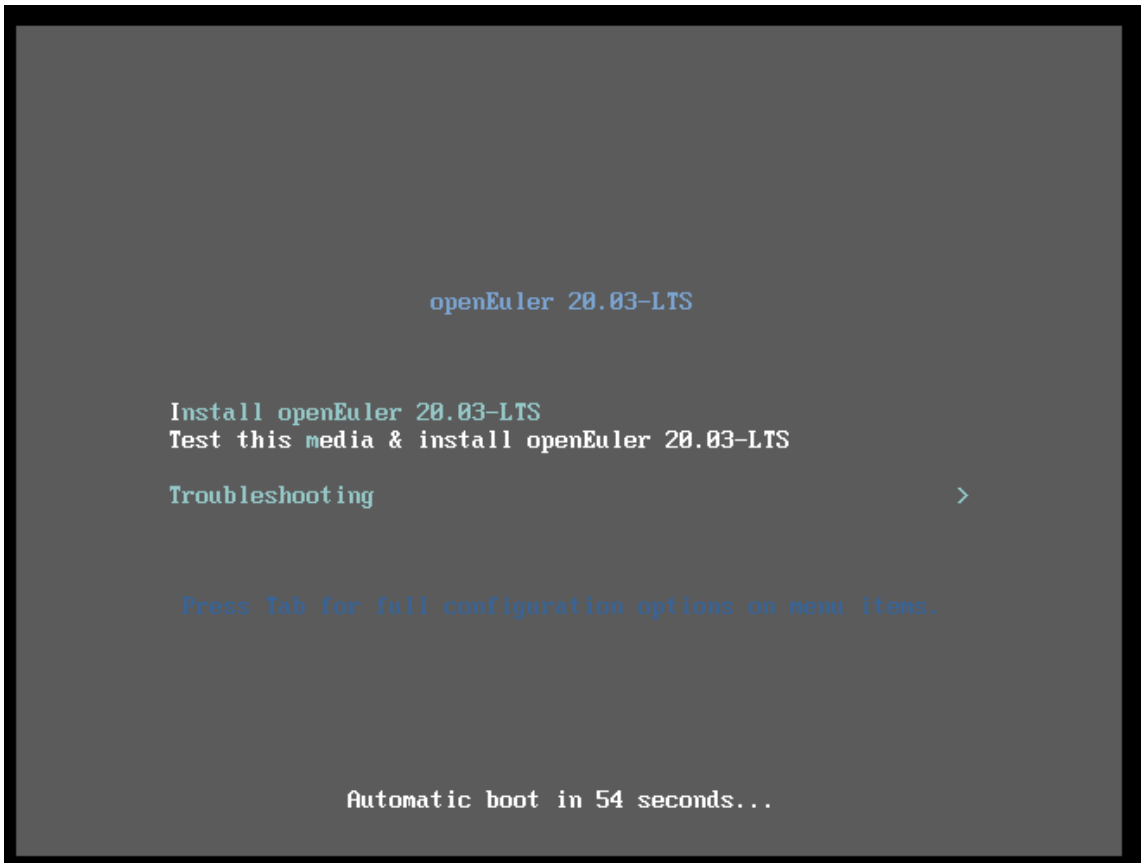

新建虚拟机向导
选择磁盘
您要使用哪个磁盘?
磁盘
创建新虚拟磁盘(V)
虚拟磁盘由主机文件系统上的一个或多个文件组成，客户机操作系统会将其视为单个硬盘。虚拟磁盘可在一台主机上或多台主机之间轻松复制或移动。
使用现有虚拟磁盘(E)
选择此选项可重新使用以前配置的磁盘。
使用物理磁盘 (适用于高级用户)(P)
选择此选项可为虚拟机提供直接访问本地硬盘的权限。需要具有管理员特权。
帮助
< 上一步(B)
下一步(N) >
取消

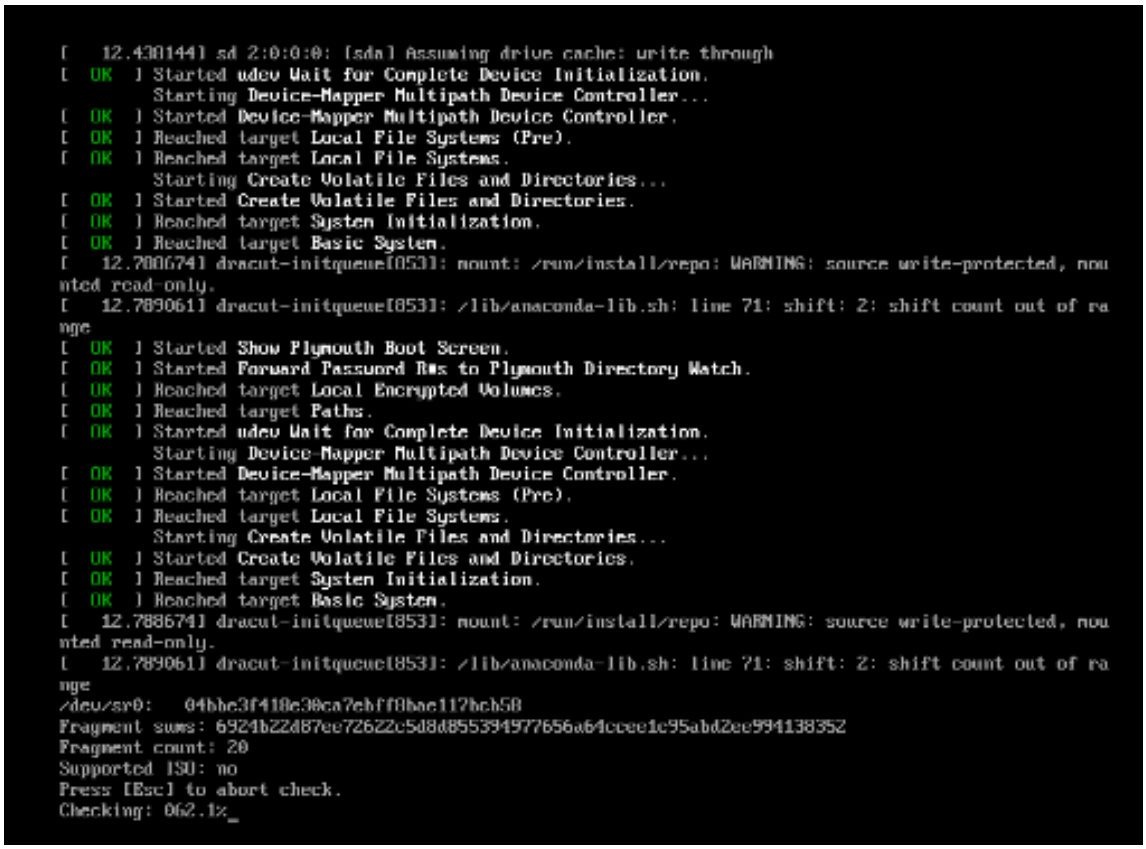

新建虚拟机向导
指定磁盘容量
磁盘大小为多少?
最大磁盘大小 (GB)(S):
45
针对 其他 Linux 4.x 内核 64 位 的建议大小: 8 GB
立即分配所有磁盘空间(A)。
分配所有容量可以提高性能，但要求所有物理磁盘空间立即可用。如果不立即分配所有空间，虚拟磁盘的空间最初很小，会随着您向其中添加数据而不断变大。
将虚拟磁盘存储为单个文件(O)
将虚拟磁盘拆分成多个文件(M)
拆分磁盘后，可以更轻松地在计算机之间移动虚拟机，但可能会降低大容量磁盘的性能。
帮助
< 上一步(B)
下一步(N) >
取消


新建虚拟机向导
指定磁盘文件
您要在何处存储磁盘文件?
磁盘文件(F)
将使用此文件名创建一个 45 GB 磁盘文件。
openEuler-20.03-LTS.vmdk
浏览(R)...
帮助
< 上一步(B)
下一步(N) >
取消


新建虚拟机向导
已准备好创建虚拟机
单击"完成"创建虚拟机。然后可以安装 其他 Linux 4.x 内核 64 位。
将使用下列设置创建虚拟机:
名称: openEuler-20.03-LTS
位置: D:\openEuler
版本: Workstation 17.x
操作系统: 其他 Linux 4.x 内核 64 位
硬盘: 45 GB
内存: 4096 MB
网络适配器: NAT
其他设备: 4 个 CPU 内核, CD/DVD, USB 控制器, 打印机, 声卡
自定义硬件(C)...
< 上一步(B)
完成
取消

②进入虚拟机，开始安装


openEuler 20.03-LTS
Install openEuler 20.03-LTS
Test this media & install openEuler 20.03-LTS
Troubleshooting
Press Tab for full configuration options on menu items.
Automatic boot in 54 seconds...


[12.430144] sd 2:0:0:0: [sda] Assuming drive cache: write through
[OK] Started udev Wait for Complete Device Initialization.
Starting Device-Mapper Multipath Device Controller...
[OK] Started Device-Mapper Multipath Device Controller.
[OK] Reached target Local File Systems (Pre).
[OK] Reached target Local File Systems.
Starting Create Volatile Files and Directories...
[OK] Started Create Volatile Files and Directories.
[OK] Reached target System Initialization.
[OK] Reached target Basic System.
[12.700674] dracut-initqueue[853]: mount: /run/install/repo: WARNING: source write-protected, mou
nted read-only.
[12.789061] dracut-initqueue[853]: /lib/anaconda-lib.sh: line 71: shift: 2: shift count out of ra
nge
[OK] Started Show Plymouth Boot Screen.
[OK] Started Forward Password R#s to Plymouth Directory Watch.
[OK] Reached target Local Encrypted Volumes.
[OK] Reached target Paths.
[OK] Started udev Wait for Complete Device Initialization.
Starting Device-Mapper Multipath Device Controller...
[OK] Started Device-Mapper Multipath Device Controller.
[OK] Reached target Local File Systems (Pre).
[OK] Reached target Local File Systems.
Starting Create Volatile Files and Directories...
[OK] Started Create Volatile Files and Directories.
[OK] Reached target System Initialization.
[OK] Reached target Basic System.
[12.788674] dracut-initqueue[853]: mount: /run/install/repo: WARNING: source write-protected, mou
nted read-only.
[12.789061] dracut-initqueue[853]: /lib/anaconda-lib.sh: line 71: shift: 2: shift count out of ra
nge
/dev/sr0: 04bbe3f418e30ca7ebff8bae117bcb58
Fragment sums: 69241b22d87ee726222c5d8d855394977656a64ccee1c95abd2ee994138352
Fragment count: 20
Supported ISO: no
Press [Esc] to abort check.
Checking: 062.1%_


软件选择
完成(D)
openEuler 20.03-LTS 安装
cn
基本环境
最小安装
基本功能。
服务器
集成的易于管理的服务器
虚拟化主机
最小虚拟化主机。
已选环境的附加选项
标准
标准安装。
容器管理
用于管理 Linux 容器的工具
开发工具
基本开发环境。
无图形终端系统管理工具
用于管理无图像终端系统的工具。
传统 UNIX 兼容性
用于从继承 UNIX 环境中迁移或者可用于该环境的兼容程序。
网络服务器
这些软件包包括基于网络的服务器，例如 DHCP、Kerberos 和 NIS。
科学记数法支持
用于数学和科学计算以及平行计算的工具。
安全性工具
用于完整性和可信验证的安全性工具。


openEuler
安装信息摘要
openEuler 20.03-LTS 安装
cn
本地化
键盘(K)
汉语
语言支持(L)
简体中文 (中国)
时间和日期(T)
亚洲/上海 时区
软件
安装源(I)
本地介质
软件选择(S)
最小安装
系统
安装位置(D)
已选择自动分区
网络和主机名(N)
有线（ens33）已连接
退出(Q)
开始安装(B)
在点击"开始安装"按钮前我们并不会操作您的磁盘。

（3）安装桌面环境（gnome）以及 terminal

①配置清华源

```
vim /etc/yum.repos.d/openEuler_x86_64.repo #打开配置文件
```

```
# 添加如下内容
[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-openEuler
```

```
#Copyright (c) [2019] Huawei Technologies Co., Ltd.
#generic-repos is licensed under the Mulan PSL v1.
#You can use this software according to the terms and conditions of the Mulan PSL v1.
#You may obtain a copy of Mulan PSL v1 at:
#    http://license.coscl.org.cn/MulanPSL
#THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, WITHOUT WARRANTIES OF ANY KIND, EITHER EXPRESS OR
#IMPLIED, INCLUDING BUT NOT LIMITED TO NON-INFRINGEMENT, MERCHANTABILITY OR FIT FOR A PARTICULAR
#PURPOSE.
#See the Mulan PSL v1 for more details.
[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
~
```

②安装 gnome、terminal

```
dnf install gnome-shell gdm gnome-session   #安装 gnome 及相关组件
dnf install gnome-terminal  #安装 terminal
#设置开机自启动
systemctl enable gdm.service
systemctl set-default graphical.target
#补全丢失文件
cd /tmp
wget https://gitee.com/name1e5s/xsession/raw/master/Xsession
mv Xsession /etc/gdm/
chmod 0777 /etc/gdm/Xsession
```

```
[buptxiangfeng2022211570@localhost ~]$ sudo dnf install gnome-shell gdm gnome-session
```

```
Installed:
  gdm-1:3.30.1-7.oe1.x86_64                                   gnome-session-3.30.1-3.oe1.x86_64                   gnome-shell-3.30.1-5.oe1.x86_64
  vino-3.22.0-12.oe1.x86_64                                   GConf2-3.2.6-24.oe1.x86_64                          accountsservice-0.6.54-2.oe1.x86_64
  accountsservice-libs-0.6.54-2.oe1.x86_64                    bolt-0.5-2.oe1.x86_64                               bubblewrap-0.3.3-1.oe1.x86_64
  cheese-libs-2:3.32.0-1.oe1.x86_64                           clutter-1.26.2-9.oe1.x86_64                         clutter-gst3-3.0.26-4.oe1.x86_64
  clutter-gtk-1.8.4-5.oe1.x86_64                              cogl-1.22.4-2.oe1.x86_64                            color-filesystem-1-14.oe1.noarch
  colord-1.4.3-6.oe1.x86_64                                   colord-gtk-0.1.26-10.oe1.x86_64                     cups-pk-helper-0.2.6-8.oe1.x86_64
  evolution-data-server-3.30.1-3.oe1.x86_64                   evolution-data-server-langpacks-3.30.1-3.oe1.noarch flac-1.3.2-12.oe1.x86_64
  freeglut-3.0.0-10.oe1.x86_64                                geocode-glib-3.26.1-4.oe1.x86_64                    gjs-1.54.1-5.oe1.x86_64
  glew-2.1.0-3.oe1.x86_64                                     gmime30-3.2.0-4.oe1.x86_64                          gnome-bluetooth-1:3.28.2-3.oe1.x86_64
  gnome-control-center-3.30.1-10.oe1.x86_64                   gnome-control-center-filesystem-3.30.1-10.oe1.noarch gnome-desktop3-3.30.1-3.oe1.x86_64
  gnome-keyring-3.28.2-3.oe1.x86_64                           gnome-keyring-pam-3.28.2-3.oe1.x86_64               gnome-online-accounts-3.30.0-5.oe1.x86_64
  gnome-settings-daemon-3.30.1.2-2.oe1.x86_64                 gnome-themes-standard-3.27.90-3.oe1.x86_64          google-noto-emoji-fonts-20180814-4.oe1.noarch
  grilo-0.3.9-3.oe1.x86_64                                    ibus-1.5.19-7.oe1.x86_64                            ibus-libs-1.5.19-7.oe1.x86_64
  iio-sensor-proxy-2.4-5.oe1.x86_64                           libXdmcp-1.1.3-2.oe1.x86_64                         libXfont2-2.0.3-4.oe1.x86_64
  libasyncns-0.8-17.oe1.x86_64                                libcanberra-0.30-25.oe1.x86_64                      libcanberra-gtk3-0.30-25.oe1.x86_64
  libdmx-1.1.4-6.oe1.x86_64                                   libevdev-1.5.9-6.oe1.x86_64                         libgdata-0.17.9-4.oe1.x86_64
  libgnomekbd-3.26.1-1.oe1.x86_64                             libgtop2-2.40.0-1.oe1.x86_64                        libgweather-3.32.2-3.oe1.x86_64
  libimobiledevice-1.2.0-18.oe1.x86_64                        libinput-1.14.1-2.oe1.x86_64                        libnma-1.8.22-2.oe1.x86_64
  liboauth-1.0.3-13.oe1.x86_64                                libplist-2.0.0-13.oe1.x86_64                        libquvi-0.9.4-16.oe1.x86_64
  libquvi-scripts-0.9.20131130-14.oe1.x86_64                  libsndfile-1.0.28-18.oe1.x86_64                     libunwind-1.3.1-3.oe1.x86_64
  libusbmuxd-1.0.10-12.oe1.x86_64                             libwacom-0.31-2.oe1.x86_64                          libwacom-data-0.31-2.oe1.x86_64
  libxkbcommon-x11-0.8.4-3.oe1.x86_64                         libxkbfile-1.1.0-2.oe1.x86_64                       libxklavier-5.4-18.oe1.x86_64
  llvm-libs-7.0.0-9.oe1.x86_64                                lua-expat-1.3.0-16.oe1.x86_64                       lua-json-1.3.2-13.oe1.noarch
  lua-lpeg-1.0.2-2.oe1.x86_64                                 lua-socket-3.0-0.19.oe1.x86_64                      mesa-demos-8.3.0-13.oe1.x86_64
  mesa-dri-drivers-18.2.2-6.oe1.x86_64                        mesa-filesystem-18.2.2-6.oe1.x86_64                 mesa-libGLU-9.0.1-1.oe1.x86_64
  mobile-broadband-provider-info-20190116-1.oe1.noarch        mozilla-filesystem-1.9-21.oe1.x86_64                mtdev-1.1.5-15.oe1.x86_64
  mutter-3.30.1-7.oe1.x86_64                                  nm-connection-editor-1.8.22-2.oe1.x86_64            pipewire-0.2.7-1.oe1.x86_64
  pulseaudio-12.2-3.oe1.x86_64                                rtkit-0.11-26.oe1.x86_64                            sbc-1.4-1.oe1.x86_64
  sound-theme-freedesktop-0.8-12.oe1.noarch                   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

Complete!
[buptxiangfeng2022211570@localhost ~]$ _
```

```
[buptxiangfeng20222115700@localhost ~]$ sudo dnf install gnome-terminal
Last metadata expiration check: 0:01:03 ago on Wed 29 May 2024 03:15:35 PM CST.
Dependencies resolved.
================================================================================================================================================
 Package                                Architecture                   Version                                Repository                      Size
================================================================================================================================================
Installing:
 gnome-terminal                         x86_64                         3.30.1-3.oe1                           oserpo                         1.3 M
Installing dependencies:
 exempi                                 x86_64                         2.4.5-4.oe1                            oserpo                         614 k
 exiv2                                  x86_64                         0.26-17.oe1                            oserpo                         1.6 M
 gnome-autoar                           x86_64                         0.2.3-4.oe1                            oserpo                          49 k
 gvfs                                   x86_64                         1.40.2-6.oe1                           oserpo                         428 k
 gvfs-client                            x86_64                         1.40.2-6.oe1                           oserpo                         735 k
 libcdio                                x86_64                         2.0.0-8.oe1                            oserpo                         154 k
 libcdio-paranoia                       x86_64                         10.2+2.0.0-2.oe1                       oserpo                          70 k
 libexif                                x86_64                         0.6.21-20.oe1                          oserpo                         344 k
 libgexiv2                              x86_64                         0.10.8-5.oe1                           oserpo                          59 k
 libgsf                                 x86_64                         1.14.43-4.oe1                          oserpo                         228 k
 libgxps                                x86_64                         0.3.1-1.oe1                            oserpo                          88 k
 libiptcdata                            x86_64                         1.0.5-1.oe1                            oserpo                          56 k
 libosinfo                              x86_64                         1.2.0-9.oe1                            oserpo                         191 k
 nautilus                               x86_64                         3.33.90-3.oe1                          oserpo                         2.6 M
 osinfo-db                              x86_64                         20180920-2.oe1                         oserpo                         165 k
 osinfo-db-tools                        x86_64                         1.2.0-3.oe1                            oserpo                          70 k
 poppler                                x86_64                         0.67.0-5.oe1                           oserpo                         1.0 M
 poppler-data                           noarch                         0.4.9-4.oe1                            oserpo                         2.1 M
 poppler-glib                           x86_64                         0.67.0-5.oe1                           oserpo                         131 k
 taglib                                 x86_64                         1.11.1-12.oe1                          oserpo                         294 k
 tracker                                x86_64                         2.1.5-3.oe1                            oserpo                         887 k
 tracker-miners                         x86_64                         2.1.5-6.oe1                            oserpo                         701 k
 vte291                                 x86_64                         0.54.1-4.oe1                           oserpo                         250 k
```

```
  Installed:
    gnome-terminal-3.30.1-3.oe1.x86_64      exempi-2.4.5-4.oe1.x86_64          exiv2-0.26-17.oe1.x86_64           gnome-autoar-0.2.3-4.oe1.x86_64
    gvfs-1.40.2-6.oe1.x86_64                gvfs-client-1.40.2-6.oe1.x86_64    libcdio-2.0.0-8.oe1.x86_64         libcdio-paranoia-10.2+2.0.0-2.oe1.x86_64
    libexif-0.6.21-20.oe1.x86_64            libgexiv2-0.10.8-5.oe1.x86_64      libgsf-1.14.43-4.oe1.x86_64        libgxps-0.3.1-1.oe1.x86_64
    libiptcdata-1.0.5-1.oe1.x86_64          libosinfo-1.2.0-9.oe1.x86_64       nautilus-3.33.90-3.oe1.x86_64      osinfo-db-20180920-2.oe1.x86_64
    osinfo-db-tools-1.2.0-3.oe1.x86_64      poppler-0.67.0-5.oe1.x86_64        poppler-data-0.4.9-4.oe1.noarch    poppler-glib-0.67.0-5.oe1.x86_64
    taglib-1.11.1-12.oe1.x86_64             tracker-2.1.5-3.oe1.x86_64         tracker-miners-2.1.5-6.oe1.x86_64  vte291-0.54.1-4.oe1.x86_64

  Complete!
  [buptxiangfeng20222115700@localhost ~]$
```

```
[buptxiangfeng20222115700@localhost ~]$ cd /tmp
[buptxiangfeng20222115700@localhost tmp]$ wget https://gitee.com/name1e5s/xsession/raw/master/Xsession
--2024-05-29 15:18:50--  https://gitee.com/name1e5s/xsession/raw/master/Xsession
Resolving gitee.com (gitee.com)... 180.76.198.77
Connecting to gitee.com (gitee.com)|180.76.198.77|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [text/plain]
Saving to: ‘Xsession’

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

2024-05-29 15:18:50 (121 MB/s) - ‘Xsession’ saved [5145]
```

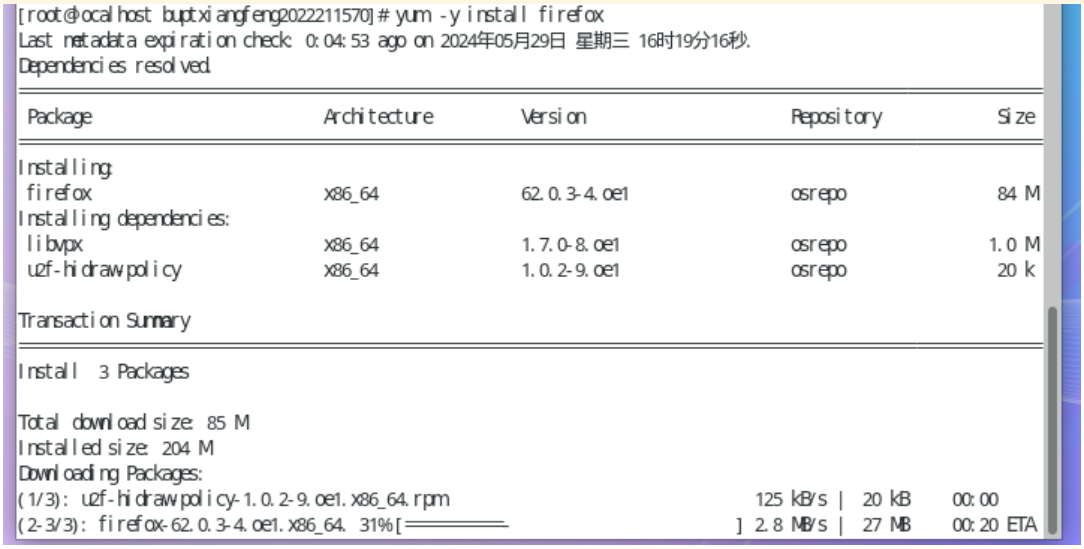
③gnome 桌面安装成功

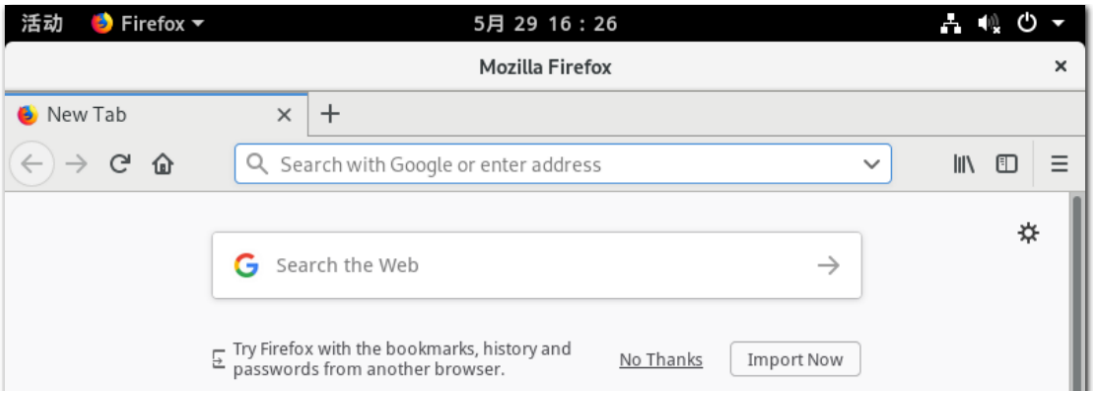
（4）执行 uname -a 指令、getconf PAGESIZE 指令

```
uname -a
getconf PAGESIZE
```

（5）安装 Firefox

```
yum -y install firefox #顺便安装firefox
```

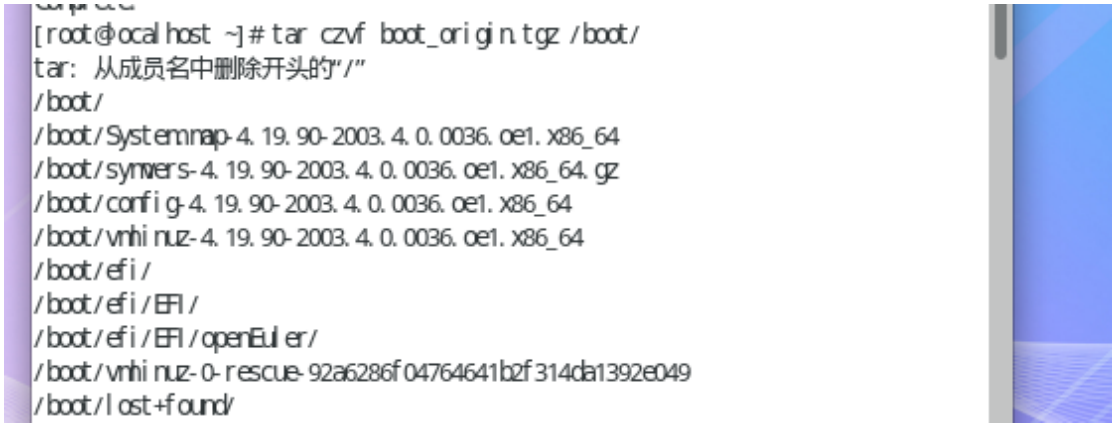


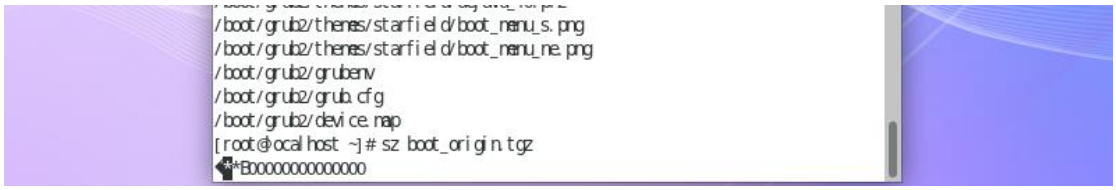


2、内核编译安装

（1）系统备份

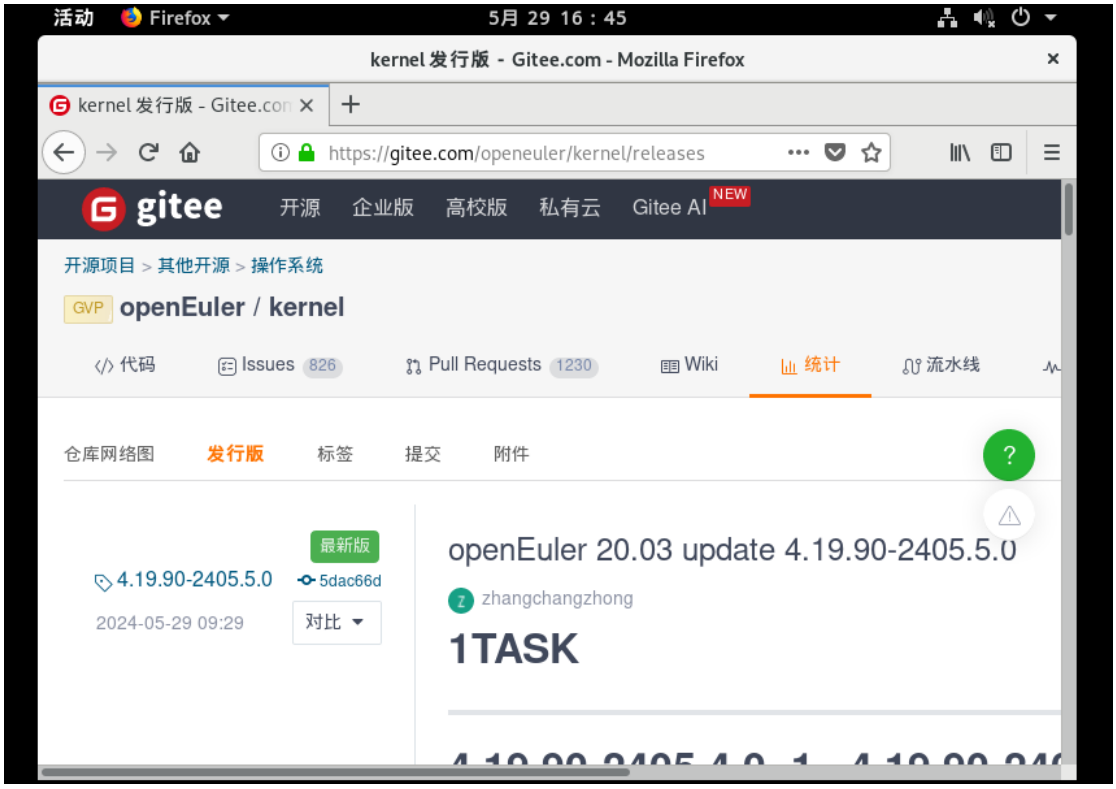
```
cd ~
dnf install lrzsz       # rz 和 sz 可以在终端下很方便的传输文件
tar czvf boot_origin.tgz /boot/
sz boot_origin.tgz     # 将备份文件发送到本地
```



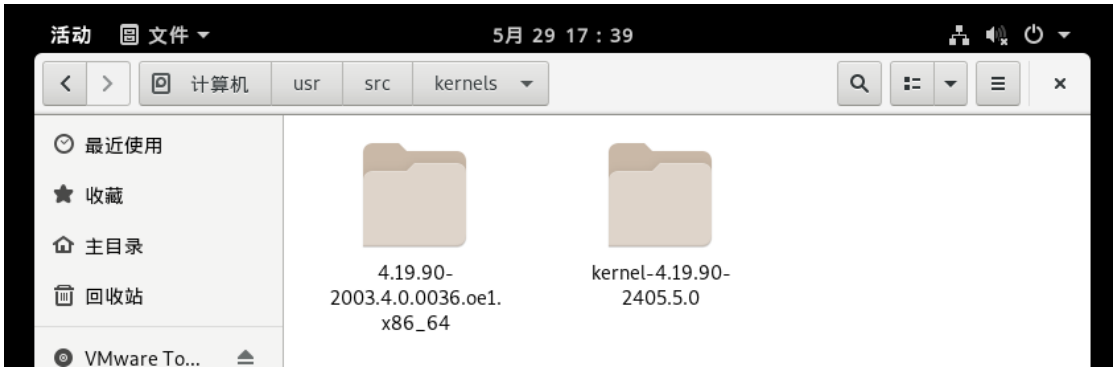


（2）内核源码下载

①在 gitee 仓库中下载 openEuler 内核压缩文件



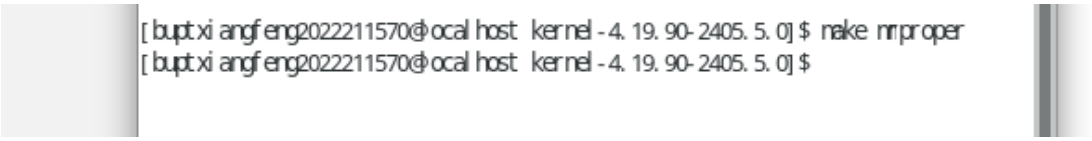
②解压缩至/usr/src/kernels



（3）清理代码树

进入解压好的源码文件夹执行命令，清理过去内核编译产生的文件。

```
make mrproper
```



（4）生成内核配置文件.config

①先将将系统原配置文件拷贝过来

```
cp -v /boot/config-4.19.90-2003.4.0.0036.oe1.x86_64.config
```

```
[buptxiangfeng2022211570@localhost ~]$ cp -v /boot/config-4.19.90-2003.4.0.0036.
oe1.x86_64 .config
'/boot/config-4.19.90-2003.4.0.0036.oe1.x86_64' -> '.config/config-4.19.90-2003.
4.0.0036.oe1.x86_64'
[buptxiangfeng2022211570@localhost ~]$
```

②执行依赖安装

```
yum install ncurses-devel
```

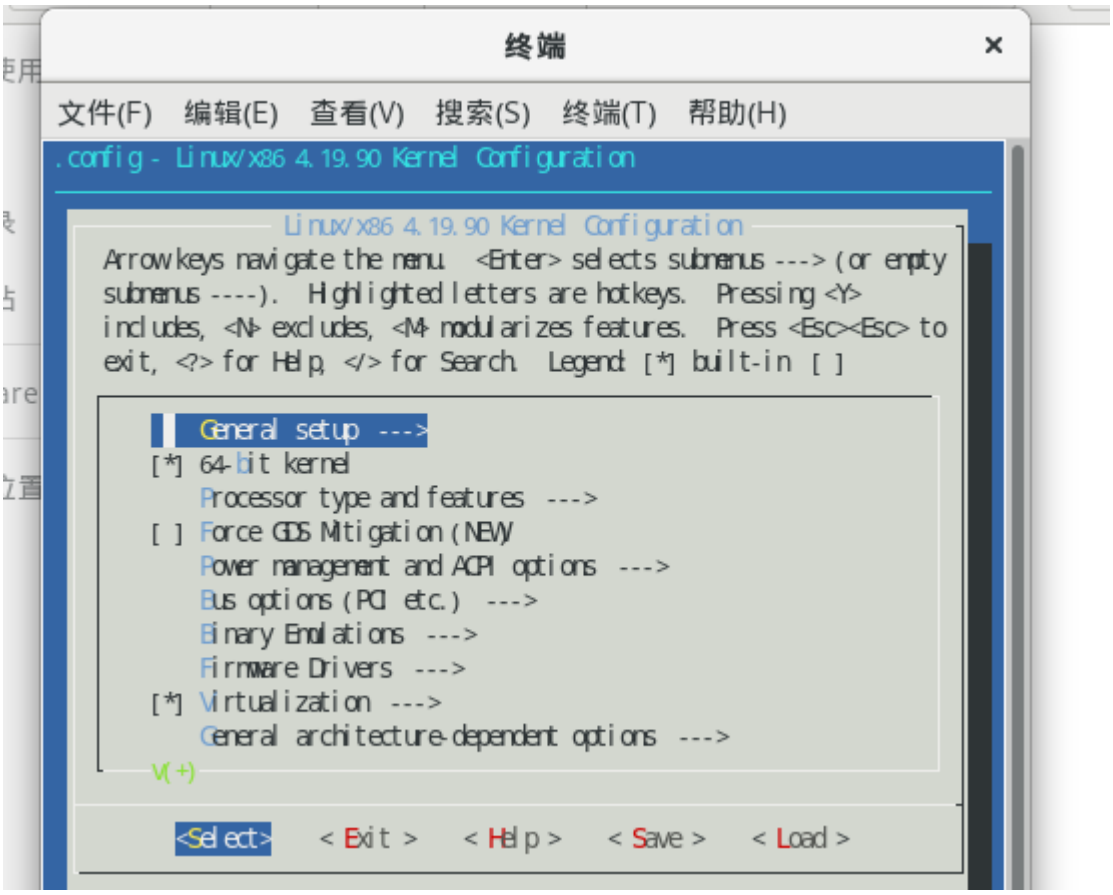
```
[root@localhost buptxiangfeng2022211570]# yum install ncurses-devel
Last metadata expiration check: 1:26:54 ago on 2024年05月29日 星期三 16时19分16
秒.
Dependencies resolved.
================================================================================
 Package          Architecture   Version          Repository      Size
================================================================================
Installing:
 ncurses-devel    x86_64         6.1-14.oe1       osrepo          643 k

Transaction Summary
================================================================================
Install  1 Package

Total download size: 643 k
Installed size: 4.7 M
Is this ok [y/N]:
```

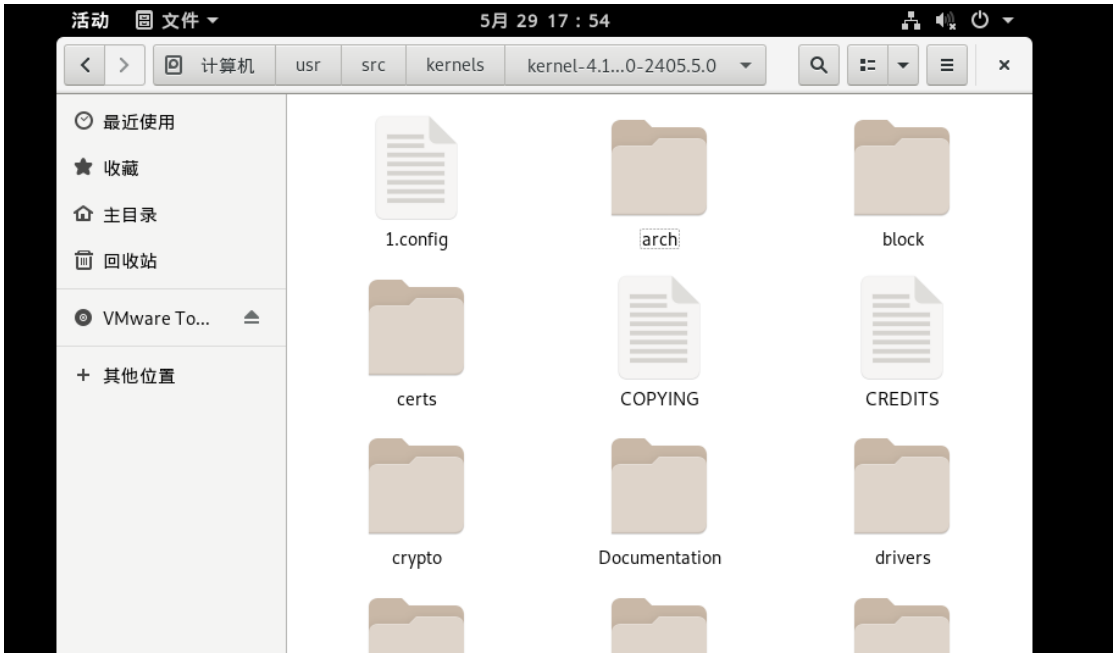
```
Install  1 Package

Total download size: 643 k
Installed size: 4.7 M
Is this ok [y/N]: y
Downloading Packages:
ncurses-devel-6.1-14.oe1.x86_64.rpm          2.3 MB/s | 643 kB     00:00
--------------------------------------------------------------------------------
Total                                        2.3 MB/s | 643 kB     00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                        1/1
  Installing       : ncurses-devel-6.1-14.oe1.x86_64                        1/1
  Running scriptlet: ncurses-devel-6.1-14.oe1.x86_64                        1/1
  Verifying        : ncurses-devel-6.1-14.oe1.x86_64                        1/1

Installed:
  ncurses-devel-6.1-14.oe1.x86_64

Complete!
[root@localhost buptxiangfeng2022211570]#
```

③对配置进行需要的更改

我没有改动直接默认配置，然后选择 Save，生成了一个.config 文件。

```
make menuconfig
```





（5）内核编译及安装

①安装所需组件

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

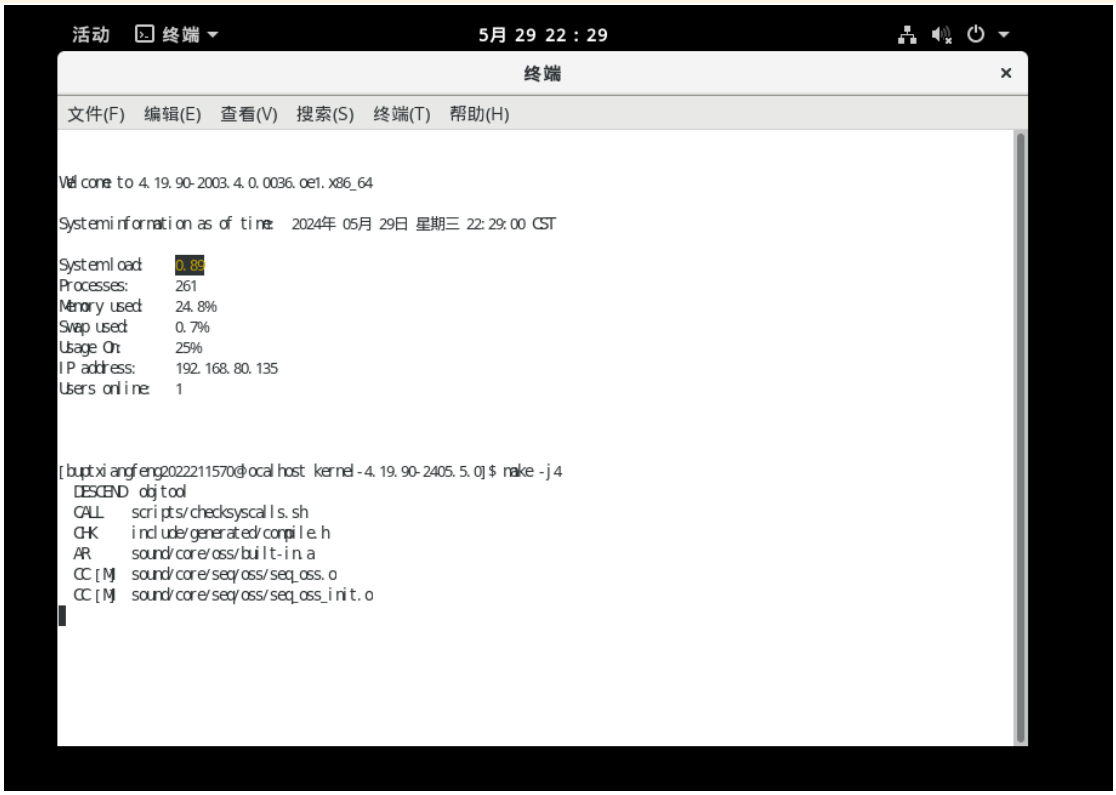
```
[root@localhost buptxiangfeng2022211570]# yum install elfutils-libelf-devel
Last metadata expiration check: 1:39:31 ago on 2024年05月29日 星期三 16时19分16秒.
Dependencies resolved.
================================================================================
 Package          Architecture  Version              Repository     Size
================================================================================
Installing:
 elfutils-devel   x86_64        0.177-3.oe1          osrepo        287 k
Installing dependencies:
 zlib-devel       x86_64        1.2.11-17.oe1        osrepo         89 k

Transaction Summary
================================================================================
Install  2 Packages

Total download size: 376 k
Installed size: 1.6 M
Is this ok [y/N]:
```

```
Last metadata expiration check: 1:40:05 ago on 2024年05月29日 星期三 16时19分16秒.
Dependencies resolved.
================================================================================
 Package          Architecture Version              Repository     Size
================================================================================
Installing:
 openssl-devel    x86_64       1:1.1.1cl-9.oe1      osrepo        1.7 M
Installing dependencies:
 e2fsprogs-devel  x86_64       1.45.3-4.oe1         osrepo        276 k
 keyutils-libs-devel x86_64    1.5.10-11.oe1        osrepo         10 k
 krb5-devel       x86_64       1.17-9.oe1           osrepo        160 k
 libselinux-devel x86_64       2.9-1.oe1            osrepo        106 k
 libsepol-devel   x86_64       2.9-1.oe1            osrepo        357 k
 libverto-devel   x86_64       0.3.1-2.oe1          osrepo         17 k
 pcre2-devel      x86_64       10.33-2.oe1          osrepo        478 k

Transaction Summary
================================================================================
Install  8 Packages

Total download size: 3.1 M
Installed size: 14 M
Is this ok [y/N]:
```

```
[root@localhost buptxiangfeng2022211570]# yum install bc
Last metadata expiration check: 1:40:31 ago on 2024年05月29日 星期三 16时19分16秒.
Package bc-1.07.1-10.oe1.x86_64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[root@localhost buptxiangfeng2022211570]#
```

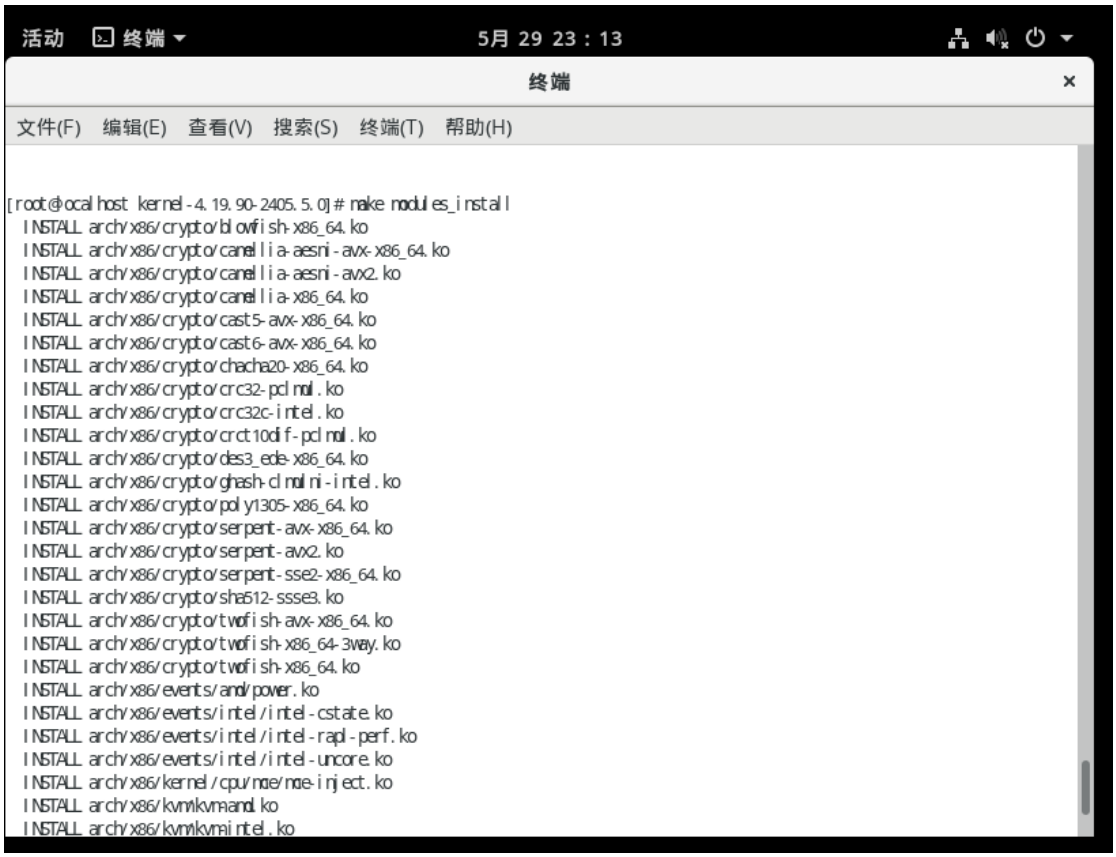
②开始编译

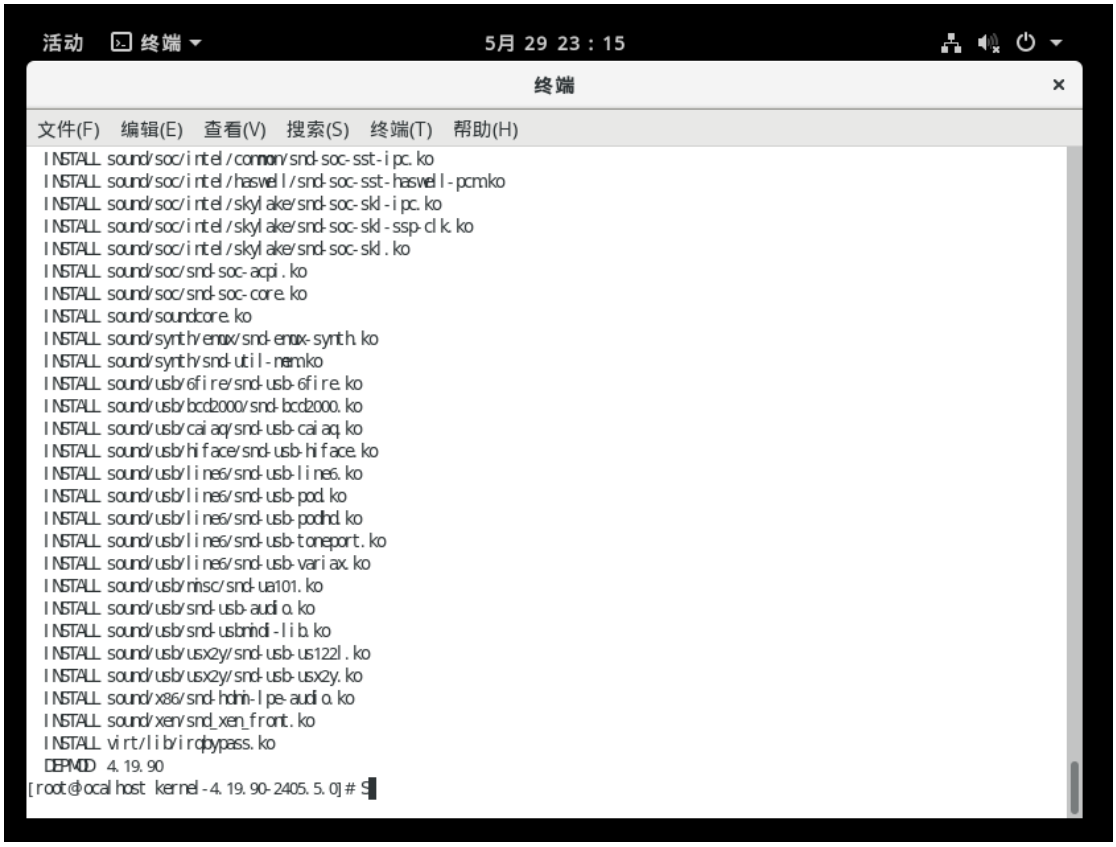
```
make -j4
```



③安装模块

```
make modules_install
```


活动 终端 5月 29 23：13
终端
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[root@ocalhost kernel-4.19.90-2405.5.0]# make modules_install
INSTALL arch/x86/crypto/blowfish-x86_64.ko
INSTALL arch/x86/crypto/camellia-aesni-avx-x86_64.ko
INSTALL arch/x86/crypto/camellia-aesni-avx2.ko
INSTALL arch/x86/crypto/camellia-x86_64.ko
INSTALL arch/x86/crypto/cast5-avx-x86_64.ko
INSTALL arch/x86/crypto/cast6-avx-x86_64.ko
INSTALL arch/x86/crypto/chacha20-x86_64.ko
INSTALL arch/x86/crypto/crc32-pclmul.ko
INSTALL arch/x86/crypto/crc32c-intel.ko
INSTALL arch/x86/crypto/crct10dif-pclmul.ko
INSTALL arch/x86/crypto/des3_ede-x86_64.ko
INSTALL arch/x86/crypto/ghash-clmulni-intel.ko
INSTALL arch/x86/crypto/poly1305-x86_64.ko
INSTALL arch/x86/crypto/serpent-avx-x86_64.ko
INSTALL arch/x86/crypto/serpent-avx2.ko
INSTALL arch/x86/crypto/serpent-sse2-x86_64.ko
INSTALL arch/x86/crypto/sha512-ssse3.ko
INSTALL arch/x86/crypto/twofish-avx-x86_64.ko
INSTALL arch/x86/crypto/twofish-x86_64-3way.ko
INSTALL arch/x86/crypto/twofish-x86_64.ko
INSTALL arch/x86/events/amd/power.ko
INSTALL arch/x86/events/intel/intel-cstate.ko
INSTALL arch/x86/events/intel/intel-rapl-perf.ko
INSTALL arch/x86/events/intel/intel-uncore.ko
INSTALL arch/x86/kernel/cpu/mce/mce-inject.ko
INSTALL arch/x86/kvm/kvm-amd.ko
INSTALL arch/x86/kvm/kvm-intel.ko


活动 终端 5月 29 23：15
终端
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
INSTALL sound/soc/intel/common/snd-soc-sst-ipc.ko
INSTALL sound/soc/intel/haswell/snd-soc-sst-haswell-pcm.ko
INSTALL sound/soc/intel/skylake/snd-soc-skl-ipc.ko
INSTALL sound/soc/intel/skylake/snd-soc-skl-ssp-clk.ko
INSTALL sound/soc/intel/skylake/snd-soc-skl.ko
INSTALL sound/soc/snd-soc-acpi.ko
INSTALL sound/soc/snd-soc-core.ko
INSTALL sound/soundcore.ko
INSTALL sound/synth/emux/snd-emux-synth.ko
INSTALL sound/synth/snd-util-mem.ko
INSTALL sound/usb/6fire/snd-usb-6fire.ko
INSTALL sound/usb/bcd2000/snd-bcd2000.ko
INSTALL sound/usb/caiaq/snd-usb-caiaq.ko
INSTALL sound/usb/hiface/snd-usb-hiface.ko
INSTALL sound/usb/line6/snd-usb-line6.ko
INSTALL sound/usb/line6/snd-usb-pod.ko
INSTALL sound/usb/line6/snd-usb-podhd.ko
INSTALL sound/usb/line6/snd-usb-toneport.ko
INSTALL sound/usb/line6/snd-usb-variax.ko
INSTALL sound/usb/misc/snd-ua101.ko
INSTALL sound/usb/snd-usb-audio.ko
INSTALL sound/usb/snd-usbmidi-lib.ko
INSTALL sound/usb/usx2y/snd-usb-us122l.ko
INSTALL sound/usb/usx2y/snd-usb-usx2y.ko
INSTALL sound/x86/snd-hdmi-lpe-audio.ko
INSTALL sound/xen/snd_xen_front.ko
INSTALL virt/lib/irqbypass.ko
DEPMOD 4.19.90
[root@ocalhost kernel-4.19.90-2405.5.0]# s

④安装内核

```
make install
```

```
[root@ocalhost kernel-4.19.90-2405.5.0]# make install
sh ./arch/x86/boot/install.sh 4.19.90 arch/x86/boot/bzImage \
        System.map "/boot"
[root@ocalhost kernel-4.19.90-2405.5.0]#
```

⑤在/boot 目录下查看新安装的内核

```
[root@ocalhost kernel-4.19.90-2405.5.0]# cd /boot/
[root@ocalhost boot]# ll
总用量 212M
-rw-r--r--. 1 root root 179K 3月 24  2020 config-4.19.90-2003.4.0.0036.oe1.x86_64
drwx-xr-x. 3 root root 4.0K 5月 29 15:48 efi
drwx------. 5 root root 4.0K 5月 29 23:17 grub2
-rw------. 1 root root  68M 5月 29 15:53 initramfs-0-rescue-92a6286f04764641b2f314da1392e049.img
-rw------. 1 root root  23M 5月 29 23:18 initramfs-4.19.90-2003.4.0.0036.oe1.x86_64.img
-rw------. 1 root root  20M 5月 29 21:16 initramfs-4.19.90-2003.4.0.0036.oe1.x86_64kdump.img
-rw------. 1 root root  72M 5月 29 23:17 initramfs-4.19.90.img
drwx-xr-x. 3 root root 4.0K 5月 29 15:52 loader
drwx------. 2 root root  16K 5月 29 15:48 lost+found
-rw-r--r--. 1 root root 326K 3月 24  2020 symvers-4.19.90-2003.4.0.0036.oe1.x86_64.gz
lrwxrwxrwx. 1 root root   24 5月 29 23:16 System.map -> /boot/System.map-4.19.90
-rw------. 1 root root 3.6M 5月 29 23:16 System.map-4.19.90
-rw-r--r--. 1 root root 3.5M 3月 24  2020 System.map-4.19.90-2003.4.0.0036.oe1.x86_64
lrwxrwxrwx. 1 root root   21 5月 29 23:16 vmlinuz -> /boot/vmlinuz-4.19.90
-rwx-xr-x. 1 root root 7.7M 5月 29 15:53 vmlinuz-0-rescue-92a6286f04764641b2f314da1392e049
-rw------. 1 root root 7.9M 5月 29 23:16 vmlinuz-4.19.90
-rwx-xr-x. 1 root root 7.7M 3月 24  2020 vmlinuz-4.19.90-2003.4.0.0036.oe1.x86_64
[root@ocalhost boot]#
```

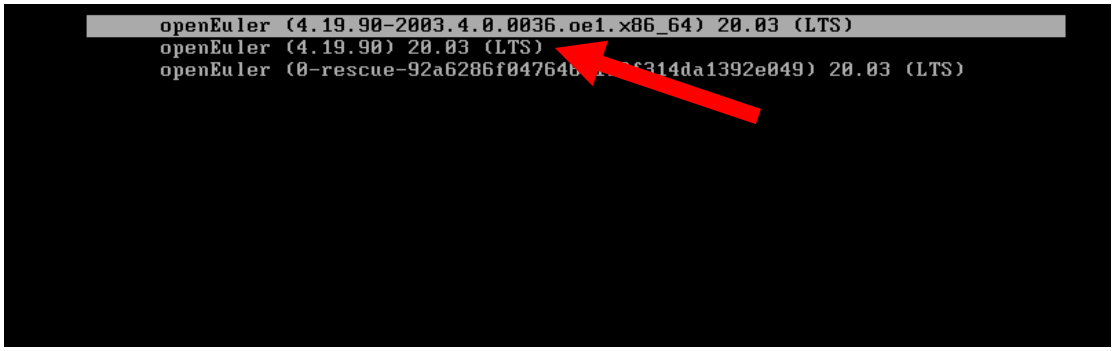
（6）更新引导

①下面的命令会根据 /boot/ 目录下的内核文件自动更新启动引导文件

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

```
[root@ocalhost boot]# grub2-mkconfig -o /boot/grub2/grub.cfg
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-4.19.90-2003.4.0.0036.oe1.x86_64
Found initrd image: /boot/initramfs-4.19.90-2003.4.0.0036.oe1.x86_64.img
Found linux image: /boot/vmlinuz-4.19.90
Found initrd image: /boot/initramfs-4.19.90.img
Found linux image: /boot/vmlinuz-0-rescue-92a6286f04764641b2f314da1392e049
Found initrd image: /boot/initramfs-0-rescue-92a6286f04764641b2f314da1392e049.img
done
[root@ocalhost boot]#
```

②重启，箭头所指即为我们新安装的内核



（7）修改默认启动内核

①查看当前系统所有可用内核

```
cat /boot/grub2/grub.cfg |grep "menuentry "
```

```
[root@ocalhost buptxiangfeng2022211570]# cat /boot/grub2/grub.cfg |grep "menuentry"
if [ x"${feature_menuentry_id}" = xy ]; then
  menuentry_id_option="--id"
  menuentry_id_option=""
export menuentry_id_option
menuentry 'openEuler (4.19.90-2003.4.0.0036.oe1.x86_64) 20.03 (LTS)' --class openeuler --class gnu-linux --class gnu --class os --u
nrestricted $menuentry_id_option 'gnulinux-4.19.90-2003.4.0.0036.oe1.x86_64-advanced-ae97c903-d856-4ce9-ac04-d41c3c17dedd' {
menuentry 'openEuler (4.19.90) 20.03 (LTS)' --class openeuler --class gnu-linux --class gnu --class os --unrestricted $menuentry_id
_option 'gnulinux-4.19.90-advanced-ae97c903-d856-4ce9-ac04-d41c3c17dedd' {
menuentry 'openEuler (0-rescue-92a6286f04764641b2f314da1392e049) 20.03 (LTS)' --class openeuler --class gnu-linux --class gnu --cla
ss os --unrestricted $menuentry_id_option 'gnulinux-0-rescue-92a6286f04764641b2f314da1392e049-advanced-ae97c903-d856-4ce9-ac04-d41c
3c17dedd' {
[root@ocalhost buptxiangfeng2022211570]#
```

②查看当前默认启动内核

```
grub2-editenv list
```

```
3c17dedd' {
[root@ocalhost buptxiangfeng2022211570]# grub2-editenv list
saved_entry=openEuler (4.19.90-2003.4.0.0036.oe1.x86_64) 20.03 (LTS)
boot_success=0
[root@ocalhost buptxiangfeng2022211570]#
```

③修改默认启动内核

```
grub2-set-default 4.19.90
```

```
[root@ocalhost buptxiangfeng2022211570]# grub2-set-default 4.19.90
[root@ocalhost buptxiangfeng2022211570]# grub2-editenv list
saved_entry=4.19.90
boot_success=0
[root@ocalhost buptxiangfeng2022211570]#
```

④执行 uname -a 指令

```
uname -a
```

```
Usage On:       53%
IP address:     192.168.80.135
Users online:   1

buptxiangfeng2022211570@ocalhost ~]$ uname -a
inux localhost.localdomain 4.19.90-2003.4.0.0036.oe1.x86_64 #1 SMP Mon Mar 23 19:10:41 UTC 2020 x86_64 x86_64 x86_64 GNU/Linux
buptxiangfeng2022211570@ocalhost ~]$
```

（二）内存管理

1、使用 kmalloc 分配 1KB，8KB 的内存，并打印指针地址

（1）创建 kmalloc.c 和 Makefile 文件

kmalloc.c

Makefile

kmalloc.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("Start kmalloc!\n");
    kmallocmem1 = (unsigned char*)kmalloc(1024, GFP_KERNEL);
    if (kmallocmem1 != NULL){
        printk(KERN_ALERT "kmallocmem1 addr = %lx\n", (unsigned
long)kmallocmem1);
    }else{
        printk("Failed to allocate kmallocmem1!\n");
    }
    kmallocmem2 = (unsigned char *)kmalloc(8192, GFP_KERNEL);
    if (kmallocmem2 != NULL){
        printk(KERN_ALERT "kmallocmem2 addr = %lx\n", (unsigned
long)kmallocmem2);
    }else{
        printk("Failed to allocate kmallocmem2!\n");
    }
    return 0;
}

static void __exit mem_module_exit(void)
{
```

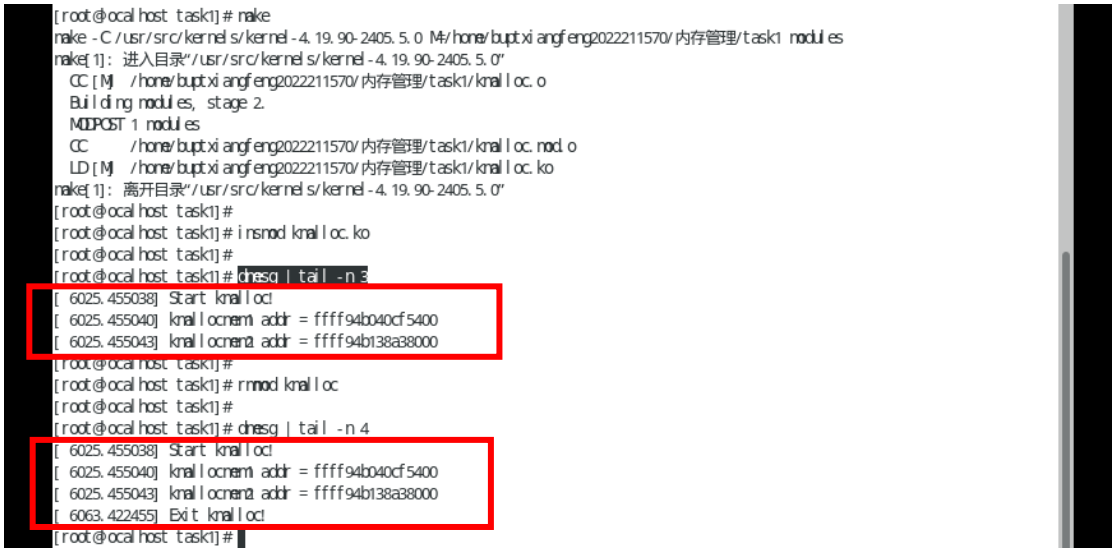
```
    kfree(kmallocmem1);
    kfree(kmallocmem2);
    printk("Exit kmalloc!\n");
}

module_init(mem_module_init);
module_exit(mem_module_exit);
```

Makefile

```
ifneq ($(KERNELRELEASE),)
    obj-m := kmalloc.o
else
    KERNELDIR ?=/usr/src/kernels/kernel-4.19.90-2405.5.0
    PWD := $(shell pwd)
default:
    $(MAKE) -C $(KERNELDIR) M=$(PWD) modules
endif
.PHONY:clean
clean:
    -rm *.mod.c *.o *.order *.symvers *.ko
```

（2）编译、加载模块

```
make

insmod kmalloc.ko

dmesg | tail -n 3

remmod kmalloc

dmesg | tail -n 4
```



（3）查看内存布局

打开/usr/src/kernels/kernel-4.19.90-2405.5.0/Documentation/x86/x86_64/mm.txt 文件。

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

0000000000000000 - 00007fffffffffff (=47 bits) user space, different per mm
hole caused by [47:63] sign extension
ffff800000000000 - ffff87ffffffffff (=43 bits) guard hole, reserved for hypervisor
ffff880000000000 - ffff887fffffffff (=39 bits) LDT remap for PTI
ffff888000000000 - ffffc87fffffffff (=64 TB) direct mapping of all phys. memory
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 ...
                                    vaddr_end for KASLR
fffffe0000000000 - fffffe7fffffffff (=39 bits) cpu_entry_area mapping
fffffe8000000000 - fffffeffffffffff (=39 bits) LDT remap for PTI
ffffff0000000000 - ffffff7fffffffff (=39 bits) %esp fixup stacks
... unused hole ...
```

```
ffffffef00000000 - fffffffeffffffff (=64 GB) EFI region mapping space
... unused hole ...
ffffffff80000000 - ffffffff9fffffff (=512 MB)  kernel text mapping, from phys 0
ffffffffa0000000 - fffffffffeffffff (1520 MB) module mapping space
[fixmap start]   - ffffffffff5fffff kernel-internal fixmap range
ffffffffff600000 - ffffffffff600fff (=4 kB) legacy vsyscall ABI
ffffffffffe00000 - ffffffffffffffff (=2 MB) unused hole

Virtual memory map with 5 level page tables:

0000000000000000 - 00ffffffffffffff (=56 bits) user space, different per mm
hole caused by [56:63] sign extension
ff00000000000000 - ff0fffffffffffff (=52 bits) guard hole, reserved for hypervisor
ff10000000000000 - ff10ffffffffffff (=48 bits) LDT remap for PTI
ff11000000000000 - ff90ffffffffffff (=55 bits) direct mapping of all phys. memory
ff91000000000000 - ff9fffffffffffff (=3840 TB) hole
ffa0000000000000 - ffd1ffffffffffff (=54 bits) vmalloc/ioremap space (12800 TB)
ffd2000000000000 - ffd3ffffffffffff (=49 bits) hole
ffd4000000000000 - ffd5ffffffffffff (=49 bits) virtual memory map (512TB)
... unused hole ...
```

```
ffdf000000000000 - fffffc0000000000 (=53 bits) kasan shadow memory (8PB)
... unused hole ...
                                    vaddr_end for KASLR
fffffe0000000000 - fffffe7fffffffff (=39 bits) cpu_entry_area mapping
... unused hole ...
ffffff0000000000 - ffffff7fffffffff (=39 bits) %esp fixup stacks
... unused hole ...
ffffffef00000000 - fffffffeffffffff (=64 GB) EFI region mapping space
... unused hole ...
ffffffff80000000 - ffffffff9fffffff (=512 MB)  kernel text mapping, from phys 0
ffffffffa0000000 - fffffffffeffffff (1520 MB) module mapping space
[fixmap start]   - ffffffffff5fffff kernel-internal fixmap range
ffffffffff600000 - ffffffffff600fff (=4 kB) legacy vsyscall ABI
ffffffffffe00000 - ffffffffffffffff (=2 MB) unused hole
```

（4）结果分析

由运行结果可知，kmalloc 分配的内存地址，位于内核空间。

2、使用 vmalloc 分别分配 8KB、1MB、64MB 的内存，打印指针地址

（1）创建 vmalloc.c 和 Makefile 文件

Makefile

vmalloc.c

vmalloc.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("Start vmalloc!\n");
    vmallocmem1 = (unsigned char*)vmalloc(8192);
    if (vmallocmem1 != NULL){
        printk("vmallocmem1 addr = %lx\n", (unsigned long)vmallocmem1);
    }else{
        printk("Failed to allocate vmallocmem1!\n");
    }
    vmallocmem2 = (unsigned char*)vmalloc(1048576);
    if (vmallocmem2 != NULL){
        printk("vmallocmem2 addr = %lx\n", (unsigned long)vmallocmem2);
    }else{
        printk("Failed to allocate vmallocmem2!\n");
    }
    vmallocmem3 = (unsigned char*)vmalloc(67108864);
    if (vmallocmem3 != NULL){
        printk("vmallocmem3 addr = %lx\n", (unsigned long)vmallocmem3);
    }else{
        printk("Failed to allocate vmallocmem3!\n");
    }
    return 0;
}
```

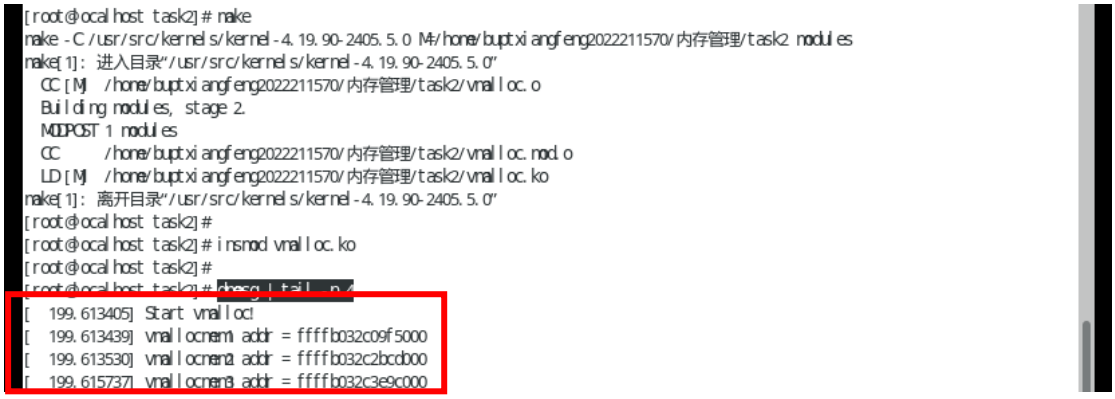
```
static void __exit mem_module_exit(void)
{
    vfree(vmallocmem1);
    vfree(vmallocmem2);
    vfree(vmallocmem3);
    printk("Exit vmalloc!\n");
}

module_init(mem_module_init);
module_exit(mem_module_exit);
```

Makefile

```
ifneq ($(KERNELRELEASE),)
    obj-m := vmalloc.o
else
    KERNELDIR ?= /usr/src/kernels/kernel-4.19.90-2405.5.0
    PWD := $(shell pwd)
default:
    $(MAKE) -C $(KERNELDIR) M=$(PWD) modules
endif
.PHONY:clean
clean:
    -rm *.mod.c *.o *.order *.symvers *.ko
```

（2）编译、加载模块

```
make

insmod vmalloc.ko

dmesg | tail -n 4

remmod vmalloc

dmesg | tail -n 5
```



```
[root@localhost task2]#
[root@localhost task2]# rmmod vmalloc
[root@localhost task2]#
[root@localhost task2]# dmesg | tail -n 5
[  199.613405] Start vmalloc!
[  199.613439] vmalloc mem1 addr = ffffb032c09f5000
[  199.613530] vmalloc mem2 addr = ffffb032c2bcd000
[  199.615737] vmalloc mem3 addr = ffffb032c3e9c000
[  227.681878] Exit vmalloc!
[root@localhost task2]#
```

（3）查看内存布局

打开/usr/src/kernels/kernel-4.19.90-2405.5.0/Documentation/x86/x86_64/mm.txt 文件。

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

0000000000000000 - 00007fffffffffff (=47 bits) user space, different per mm
hole caused by [47:63] sign extension
ffff800000000000 - ffff87ffffffffff (=43 bits) guard hole, reserved for hypervisor
ffff880000000000 - ffff887fffffffff (=39 bits) LDT remap for PTI
ffff888000000000 - ffffc87fffffffff (=64 TB) direct mapping of all phys. memory
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 ...
                                    vaddr_end for KASLR
fffffe0000000000 - fffffe7fffffffff (=39 bits) cpu_entry_area mapping
fffffe8000000000 - fffffeffffffffff (=39 bits) LDT remap for PTI
ffffff0000000000 - ffffff7fffffffff (=39 bits) %esp fixup stacks
... unused hole ...
ffffffef00000000 - fffffffeffffffff (=64 GB) EFI region mapping space
```

```
ffffffef00000000 - fffffffeffffffff (=64 GB) EFI region mapping space
... unused hole ...
ffffffff80000000 - ffffffff9fffffff (=512 MB)  kernel text mapping, from phys 0
ffffffffa0000000 - fffffffffeffffff (1520 MB) module mapping space
[fixmap start]   - ffffffffff5fffff kernel-internal fixmap range
ffffffffff600000 - ffffffffff600fff (=4 kB) legacy vsyscall ABI
ffffffffffe00000 - ffffffffffffffff (=2 MB) unused hole

Virtual memory map with 5 level page tables:

0000000000000000 - 00ffffffffffffff (=56 bits) user space, different per mm
hole caused by [56:63] sign extension
ff00000000000000 - ff0fffffffffffff (=52 bits) guard hole, reserved for hypervisor
ff10000000000000 - ff10ffffffffffff (=48 bits) LDT remap for PTI
ff11000000000000 - ff90ffffffffffff (=55 bits) direct mapping of all phys. memory
ff91000000000000 - ff9fffffffffffff (=3840 TB) hole
ffa0000000000000 - ffd1ffffffffffff (=54 bits) vmalloc/ioremap space (12800 TB)
ffd2000000000000 - ffd3ffffffffffff (=49 bits) hole
ffd4000000000000 - ffd5ffffffffffff (=49 bits) virtual memory map (512TB)
... unused hole ...
```

```
ffffffffa0000000 - fffffffffeffffff (1520 MB) module mapping space
[fixmap start]   - ffffffffff5fffff kernel-internal fixmap range
ffffffffff600000 - ffffffffff600fff (=4 kB) legacy vsyscall ABI
ffffffffffe00000 - ffffffffffffffff (=2 MB) unused hole
```

（4）结果分析

由运行结果可知，vmalloc 分配的内存地址，位于内核空间。

（三）内核时间管理

1、调用内核时钟接口打印当前时间

（1）创建 current_time.c 和 Makefile 文件

current_time.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("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("Current time: %d-%02d-%02d %02d:%02d:%02d\n", year, mon,
day, hour, min, sec);
    return 0;
}

static void __exit currenttime_exit(void)
{
    printk("Exit current_time module…\n");
```

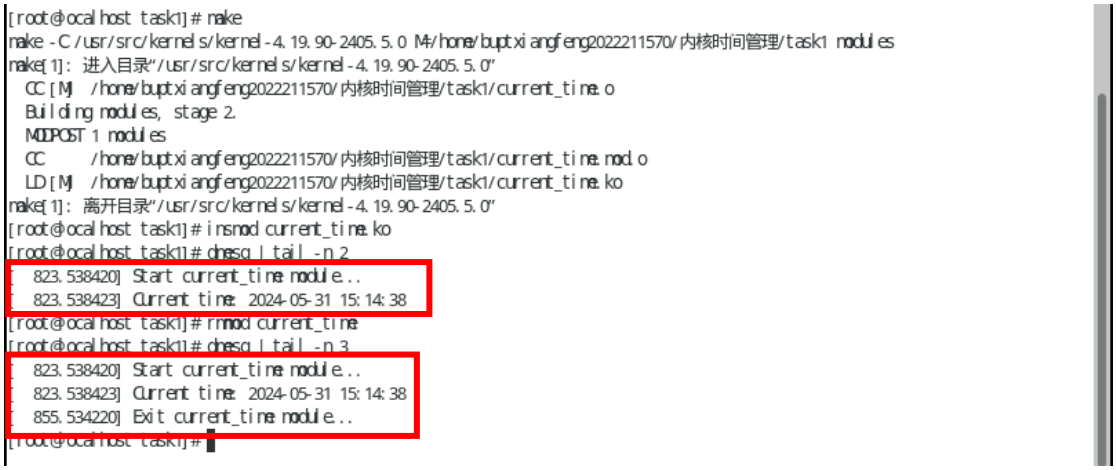
```
}

module_init(currenttime_init);
module_exit(currenttime_exit);
```

Makefile

```
ifneq ($(KERNELRELEASE),)
    obj-m := current_time.o
else
    KERNELDIR ?= /usr/src/kernels/kernel-4.19.90-2405.5.0
    PWD := $(shell pwd)
default:
    $(MAKE) -C $(KERNELDIR) M=$(PWD) modules
endif
.PHONY:clean
clean:
    -rm *.mod.c *.o *.order *.symvers *.ko
```

（2）编译运行

```
make

insmod current_time.ko

dmesg | tail -n 2

remmod current_time

dmesg | tail -n 3
```



（3）结果分析

成功在在屏幕上打印出格式化的时间、日期，并正确地加载和卸载。

2、编写 timer，在特定时刻打印 hello,world

（1）创建 timer_example.c 和 Makefile 文件

Makefile

timer_example.c

timer_example.c

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

MODULE_LICENSE("GPL");

struct timer_list timer;

void print(struct timer_list *timer)
{
    printk("hello,world!\n");
}

static int __init timer_init(void)
{
    printk("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("Exit timer_example module...\n");
}

module_init(timer_init);
module_exit(timer_exit);
```

Makefile

```
ifneq ($(KERNELRELEASE),)
    obj-m := timer_example.o
else
    KERNELDIR ?= /usr/src/kernels/kernel-4.19.90-2405.5.0
    PWD := $(shell pwd)
default:
    $(MAKE) -C $(KERNELDIR) M=$(PWD) modules
endif
.PHONY:clean
clean:
    -rm *.mod.c *.o *.order *.symvers *.ko
```

（2）编译运行

```
make

insmod timer_example.ko

dmesg | tail -n 2

dmesg -t | tail -n 2

dmesg -T | tail -n 2

remmod timer_example

dmesg -T | tail -n 3
```

```
[root@ocalhost task2]# make
make -C /usr/src/kernels/kernel-4.19.90-2405.5.0 M=/home/buptxiangfeng2022211570/内核时间管理/task2 modules
make[1]: 进入目录"/usr/src/kernels/kernel-4.19.90-2405.5.0"
  CC [M]  /home/buptxiangfeng2022211570/内核时间管理/task2/timer_example.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/buptxiangfeng2022211570/内核时间管理/task2/timer_example.mod.o
  LD [M]  /home/buptxiangfeng2022211570/内核时间管理/task2/timer_example.ko
make[1]: 离开目录"/usr/src/kernels/kernel-4.19.90-2405.5.0"
[root@ocalhost task2]#
[root@ocalhost task2]# insmod timer_example.ko
[root@ocalhost task2]#
[root@ocalhost task2]# dmesg | tail -n 2
[  232.823224] Start timer_example module...
[  243.031494] hello,world!
[root@ocalhost task2]# dmesg -t | tail -n 2
Start timer_example module...
hello,world!
[root@ocalhost task2]# dmesg -T | tail -n 2
[五 5月 31 15:33:19 2024] Start timer_example module...
[五 5月 31 15:33:30 2024] hello,world!
[root@ocalhost task2]# rmmod timer_example
[root@ocalhost task2]# dmesg -T | tail -n 3
[五 5月 31 15:33:19 2024] Start timer_example module...
[五 5月 31 15:33:30 2024] hello,world!
[五 5月 31 15:34:54 2024] Exit timer_example module...
[root@ocalhost task2]#
```

（3）结果分析

加载该内核模块 10 秒后打印 “hello,world!”，因为定时器执行了定时操作。合理使用定时器，可以使工作在指定时间点上执行，我们只需要执行一些初始化工作，设置一个超时时间，指定超时发生后执行的函数，然后激活定时器就可以了。指定的函数在定时器到期时自动执行。

3、调用内核时钟接口，监控累加计算代码的运行时间

（1）创建 sum_time.c 和 Makefile 文件

Makefile

sum_time.c

sum_time.c

```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("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("Start sum_time module...\n");
    do_gettimeofday(&tv);
    start = (int)tv.tv_sec;
    start_u = (int)tv.tv_usec;
    printk("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("The end time is: %d s %d us \n", end, end_u);
    time_cost = (end - start) * 1000000 + end_u - start_u;
    printk("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("Exit sum_time module...\n");
}

module_init(sum_init);
module_exit(sum_exit);
```

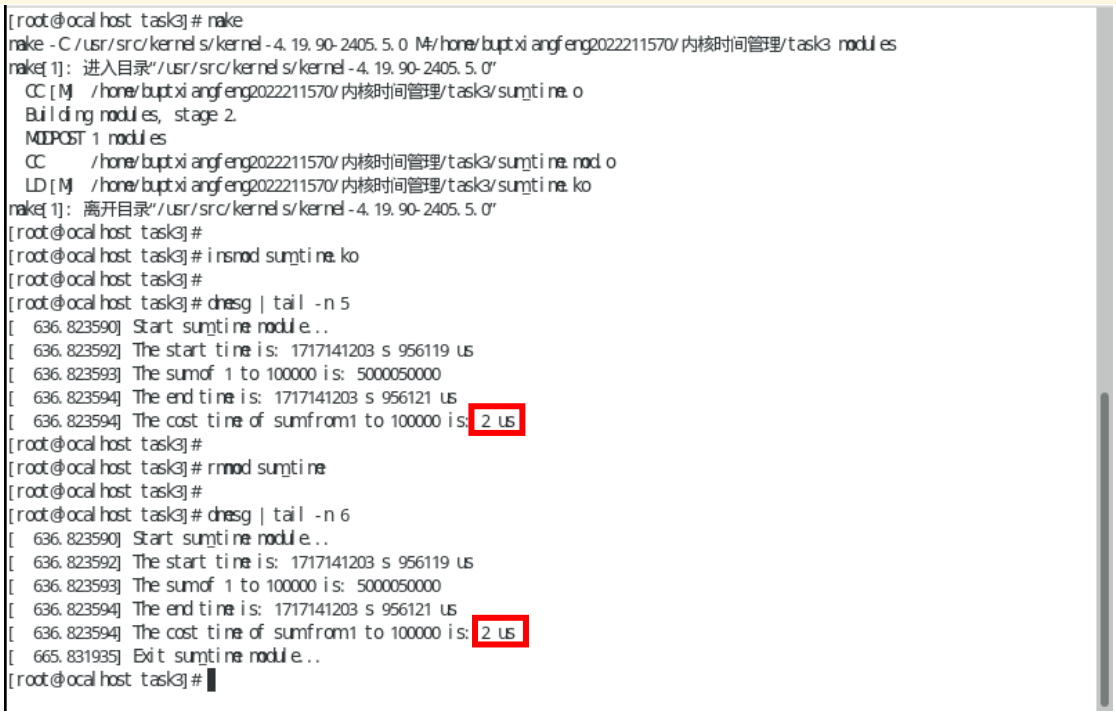
Makefile

```
ifneq ($(KERNELRELEASE),)
    obj-m := sum_time.o
else
    KERNELDIR ?= /usr/src/kernels/kernel-4.19.90-2405.5.0
    PWD := $(shell pwd)
default:
    $(MAKE) -C $(KERNELDIR) M=$(PWD) modules
endif
.PHONY:clean
```

```
clean:
    -rm *.mod.c *.o *.order *.symvers *.ko
```

（2）编译运行

```
make

insmod sum_time.ko

dmesg | tail -n 5

remmod sum_time

dmesg | tail -n 6
```



```
[root@ocalhost task3]# make
make -C /usr/src/kernels/kernel-4.19.90-2405.5.0 M=/home/buptxiangfeng2022211570/内核时间管理/task3 modules
make[1]: 进入目录"/usr/src/kernels/kernel-4.19.90-2405.5.0"
  CC [M]  /home/buptxiangfeng2022211570/内核时间管理/task3/sumtime.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/buptxiangfeng2022211570/内核时间管理/task3/sumtime.mod.o
  LD [M]  /home/buptxiangfeng2022211570/内核时间管理/task3/sumtime.ko
make[1]: 离开目录"/usr/src/kernels/kernel-4.19.90-2405.5.0"
[root@ocalhost task3]#
[root@ocalhost task3]# insmod sumtime.ko
[root@ocalhost task3]#
[root@ocalhost task3]# dmesg | tail -n 5
[  636.823590] Start sumtime module...
[  636.823592] The start time is: 1717141203 s 956119 us
[  636.823593] The sum of 1 to 100000 is: 5000050000
[  636.823594] The end time is: 1717141203 s 956121 us
[  636.823594] The cost time of sum from 1 to 100000 is: 2 us
[root@ocalhost task3]#
[root@ocalhost task3]# rmmod sumtime
[root@ocalhost task3]#
[root@ocalhost task3]# dmesg | tail -n 6
[  636.823590] Start sumtime module...
[  636.823592] The start time is: 1717141203 s 956119 us
[  636.823593] The sum of 1 to 100000 is: 5000050000
[  636.823594] The end time is: 1717141203 s 956121 us
[  636.823594] The cost time of sum from 1 to 100000 is: 2 us
[  665.831935] Exit sumtime module...
[root@ocalhost task3]#
```

（3）结果分析

由程序运行结果可以看出，从 1 到 100000 的累加和所花时间是 2 us。

四、问题及解决方案

问题	解决方案
安装 gnome 后，登录后无法进入桌面，重新安装后，可正常进入。	在 openEuler 上安装桌面环境 - 知乎 (zhihu.com) 实验一、openEuler 操作系统安装与内核编译_安装 openeuler 操作系统实验报告-CSDN 博客

内存管理、内核时间管理代码参考	NWPU_OS 教学/OpenEuler_实验 (gitee.com)
实验过程参考	OpenEuler 实验_本次实验服务器已完成内核编译(openeuler 4.19.08),可直接开始实验 -CSDN 博客

五、实验总结

此次实验，不仅增强了我的技能，也使我对 openEuler 操作系统的安装与编译、内存管理以及内核时间管理有了较为深刻的理解，为我未来的学习和工作奠定了坚实基础。