系统环境搭建与基本操作的熟悉 【本实验中 ISO 文件的下载与虚拟机镜像的生成比较消耗时间，需要约一个半小时，其余内容较快】

1.openEuler 简介

openEuler 是一款开源操作系统。当前 openEuler 内核源于 Linux，支持鲲鹏及其它多种处理器，能够充分释放计算芯片的潜能，是由全球开源贡献者构建的高效、稳定、安全的开源操作系统，适用于数据库、大数据、云计算、人工智能等应用场景。同时，openEuler 是一个面向全球的操作系统开源社区，通过社区合作，打造创新平台，构建支持多处理器架构、统一和开放的操作系统，推动软硬件应用生态繁荣发展。

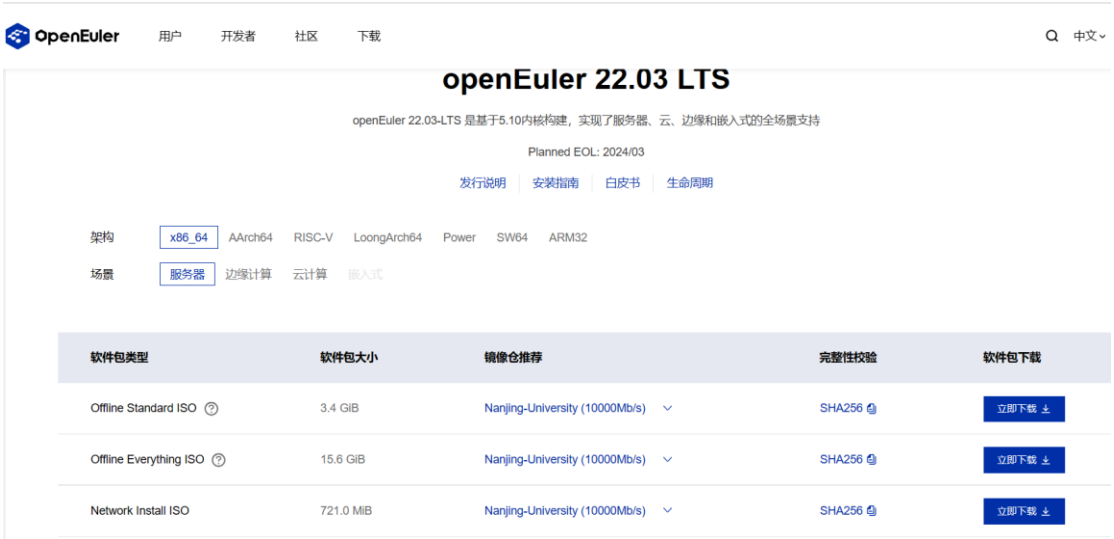
2.安装 openEuler 操作系统

openEuler 的操作系统镜像在“https://openeuler.org/zh/”下载，进入官方网站后点击链接“下载”，选择社区发行版。

点击 openEurer 22.03LTS 版本下“前往下载”链接，进入下载目录：

如果要下载 20.03 版本，翻页找到 openEuler 20.03 LTS 即可。云环境目前仅支持到 20.03LTS，本指导书安装 22.03LTS 版本。

进入下载目录后可以选择不同架构下的 OpenEuler 镜像文件，在 X86_64 下有三个文件，分别是最小标准安装，全量安装和网络安装，我们选择最小标准安装。



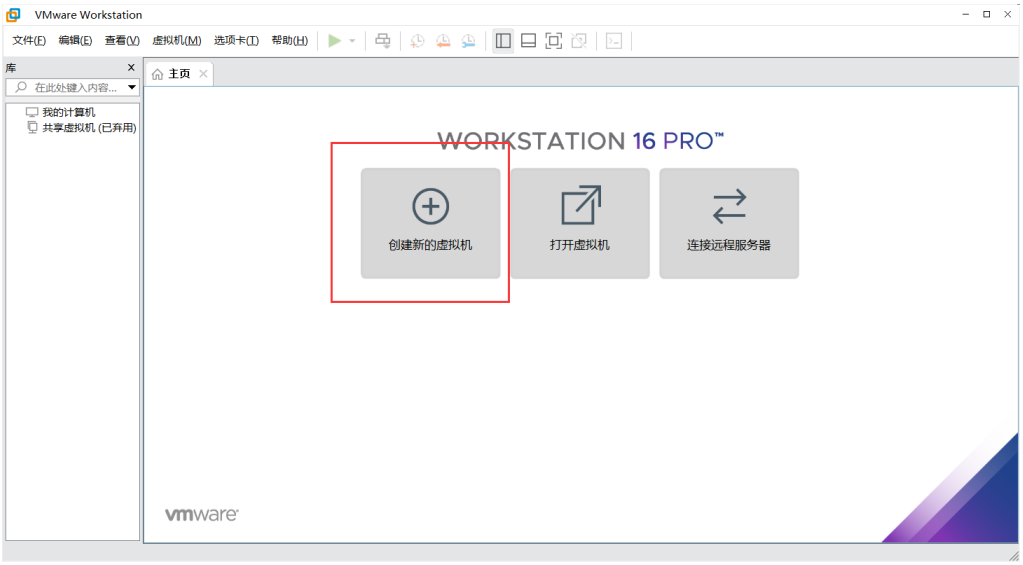
我们的个人电脑硬件基本都基于 x86_64 指令集架构设计，因此我们进入“x86_64”文件夹，该文件夹里放的是 x86_64 架构下 openEuler 的 ISO 和虚拟机镜像。

各个文件的概述如下表所示：

名称	描述
Offline Standard ISO	x86_64 架构的基础安装 ISO，包含了运行最小系统的核心组件
Offline Everything ISO	x86_64 架构的全量安装 ISO，包含了运行完整系统所需的全部组件
Network Install ISO	网络安装 ISO

校验文件的使用方法我们可以参考官网，如果校验值一致说明 iso 文件完整性没有破坏，如果校验值不一致则可以确认文件完整性已被破坏，需要重新获取。

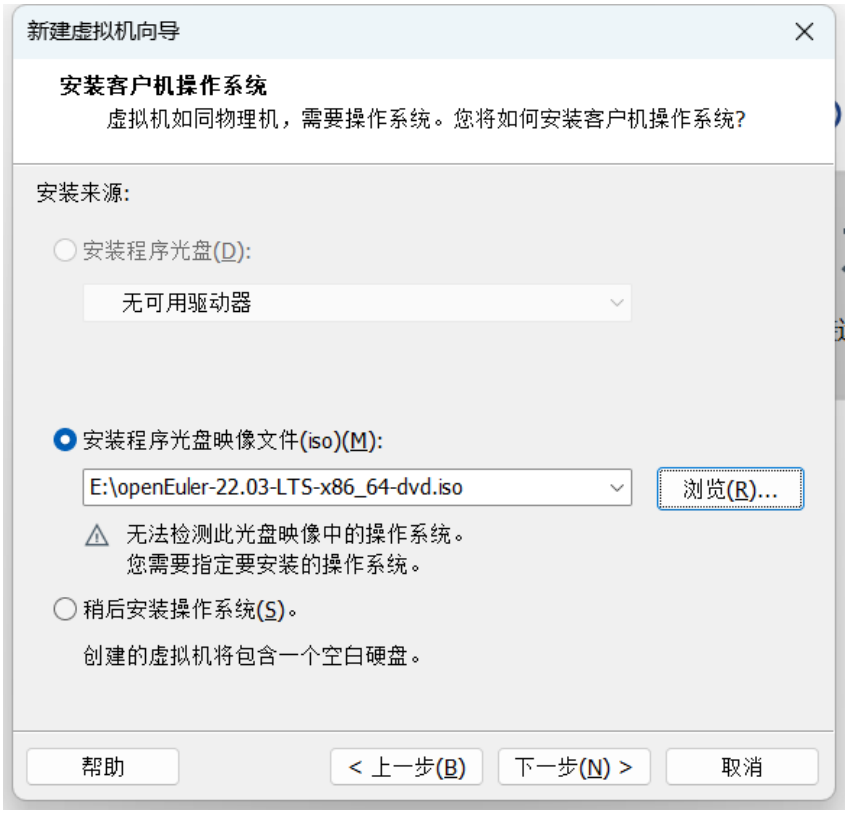
文件下载完毕后打开 Vmware Workstation，在其主页选择“创建新的虚拟机”：



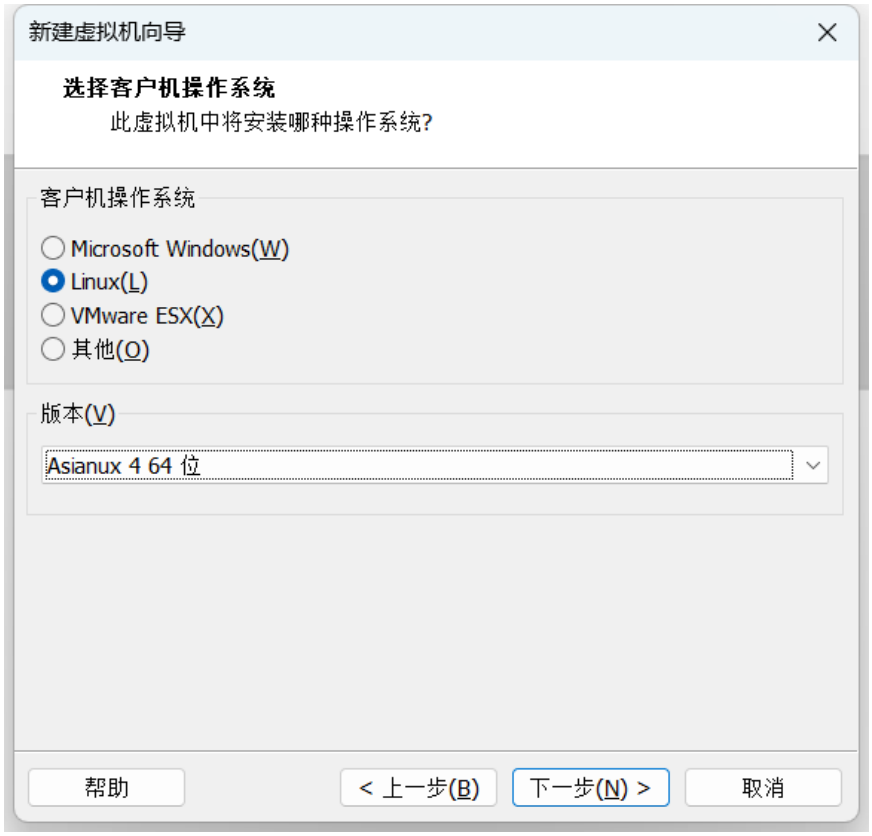
openEuler 系统的设计基于 Linux 5.10 版本的内核，由于该操作系统还比较年轻，尚未成为主流，VMware 没有提供自动化的配置选项，因此选择自定义（高级）配置：

兼容性方面不需要进行设置，直接进入下一步：

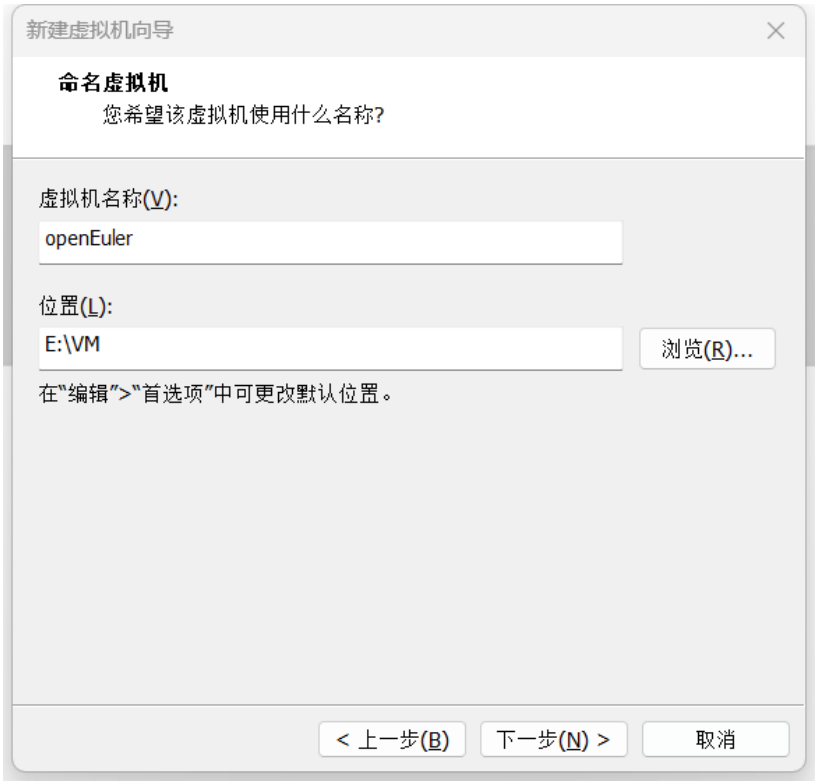
选择 ISO 文件所在的文件目录，这里的“无法检测”正是因为 VMware 没有提供自动化的配置选项，不会影响配置流程，直接进入下一步：



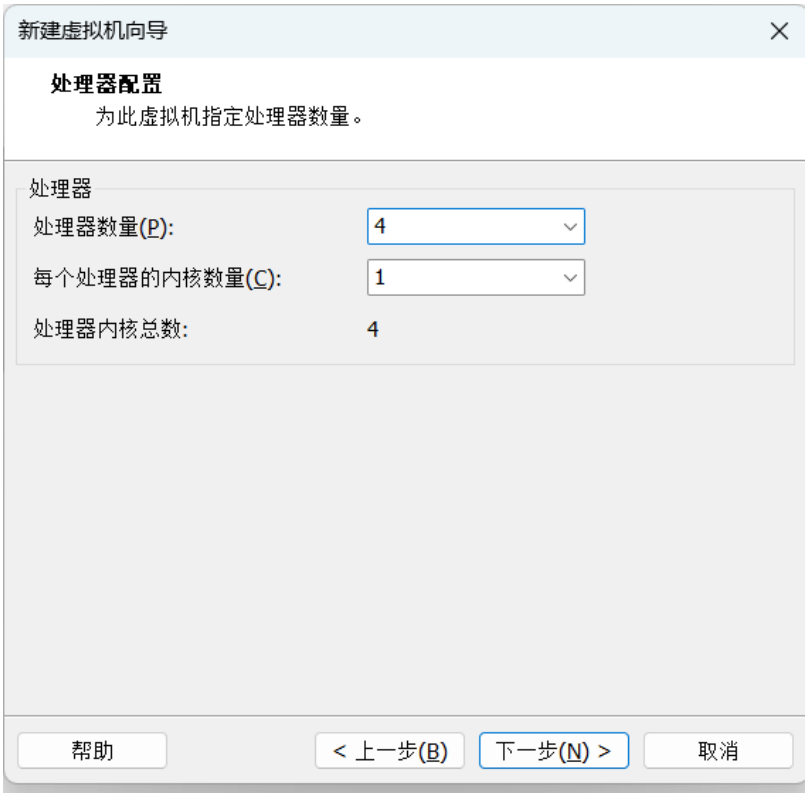
openEuler 系统的设计基于 Linux 5.10 版本的内核，因此选择“其它 Linux 5.x 或更高版本内核 64 位”：



接下来命名虚拟机，并指定虚拟机相关文件的存储位置：



对虚拟机所使用的“处理器”进行设置：

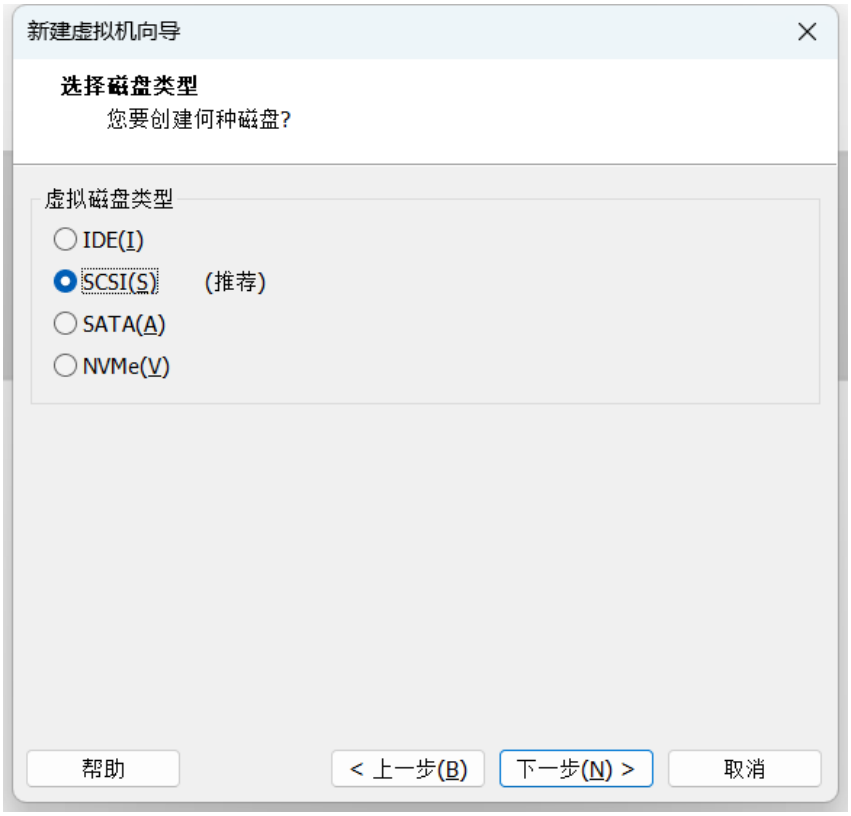


对虚拟机所使用的“内存”进行设置：

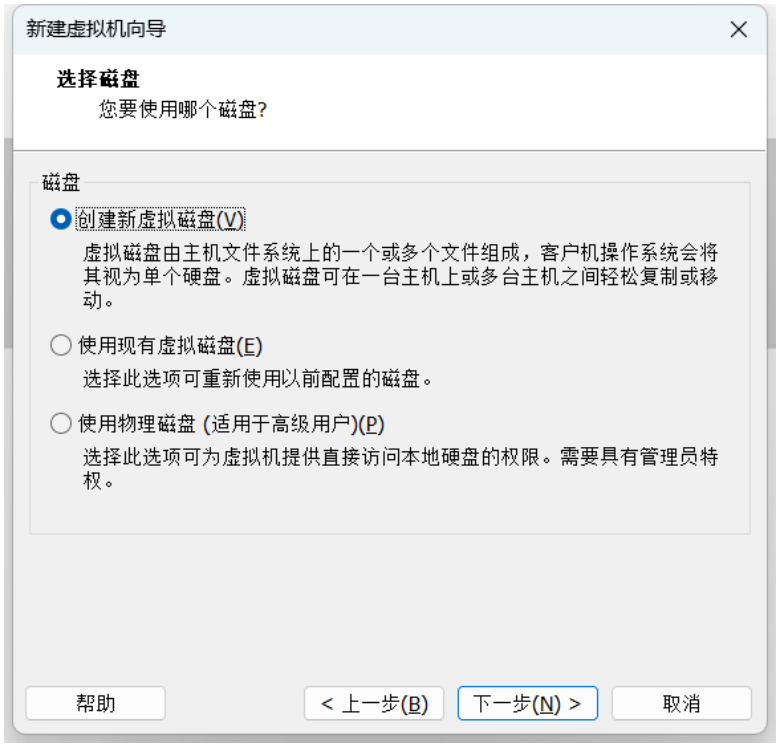
对虚拟机的网络配置进行设置：

对虚拟机的 I/O 进行设置：

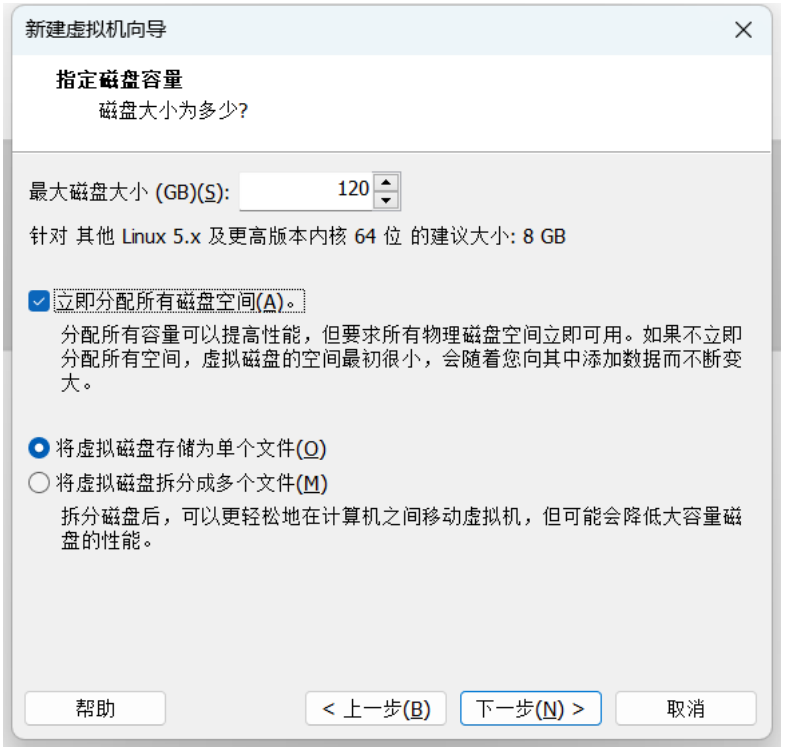
对虚拟机所使用的“外存（磁盘）”进行设置：



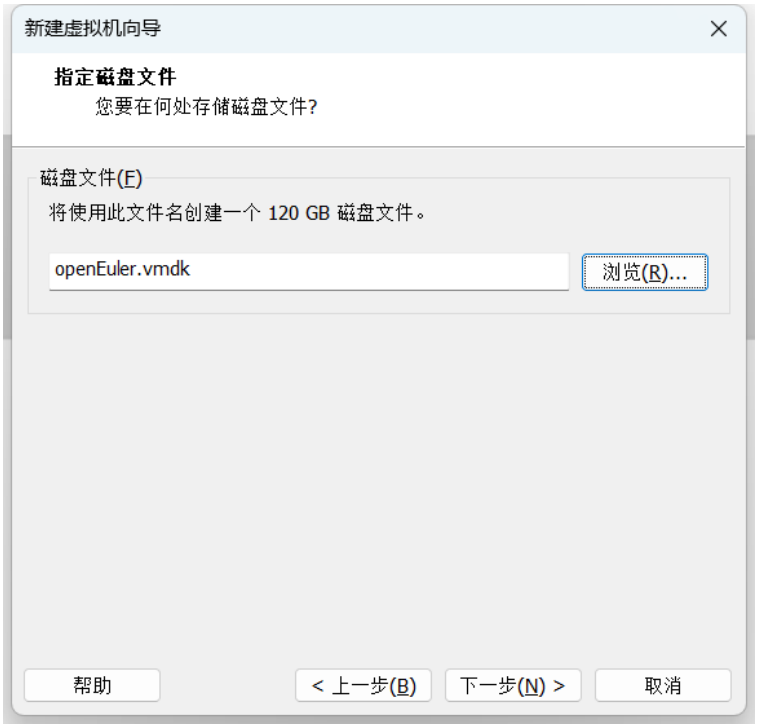
为虚拟机使用的“外存”分配存储空间：



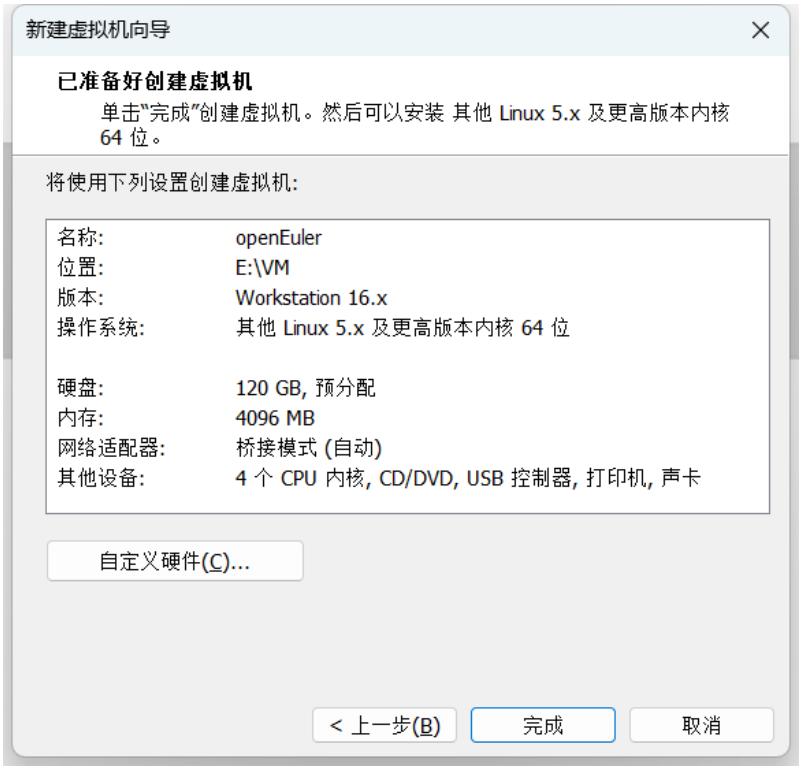
根据官网文档分配 120G 磁盘空间给虚拟机，为读写性能考虑选择将虚拟磁盘存储为单个文件：



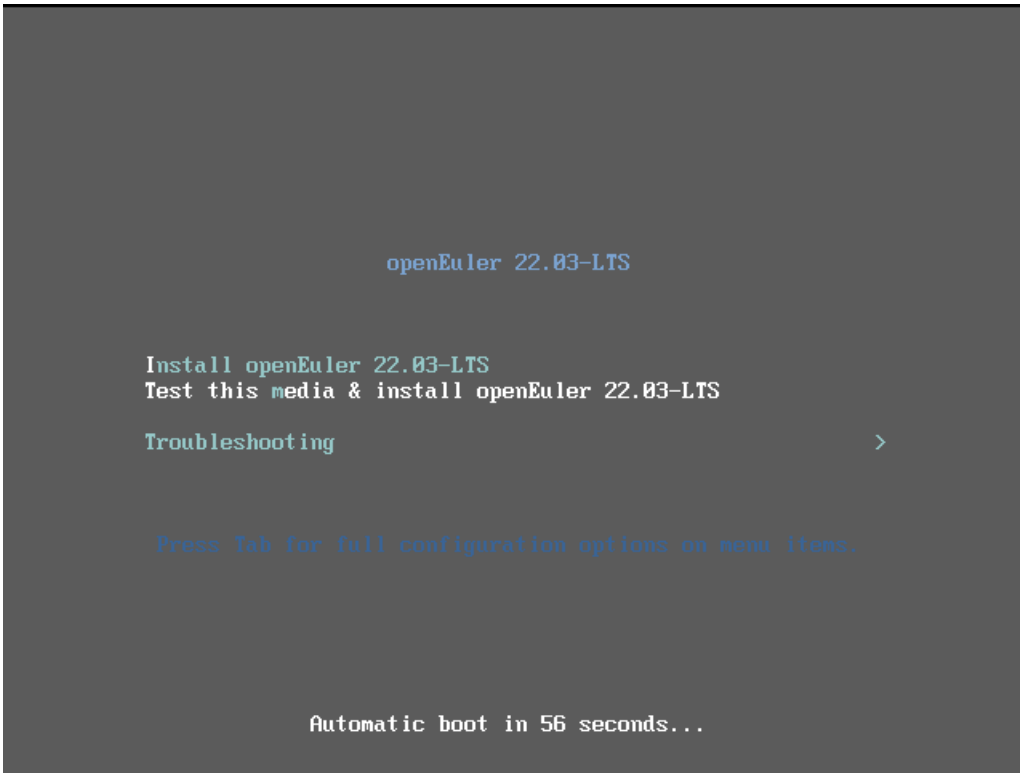
选择该虚拟磁盘文件的存储路径：

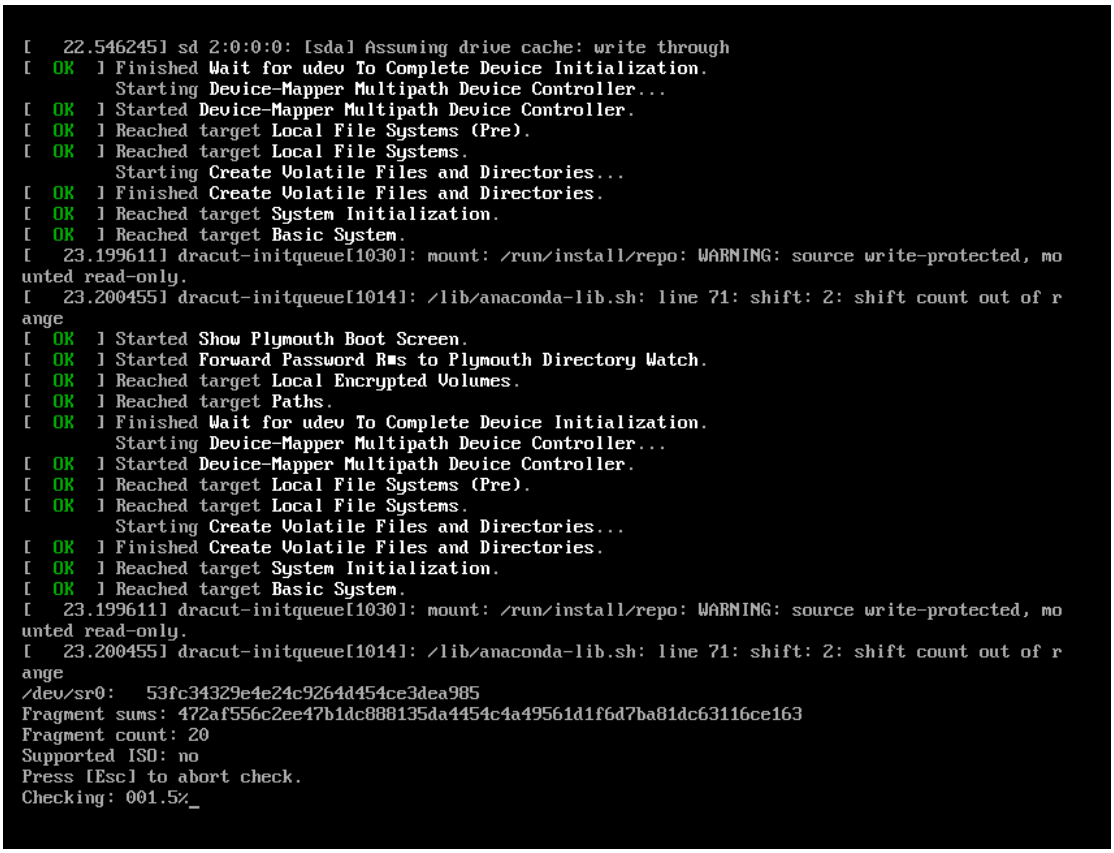


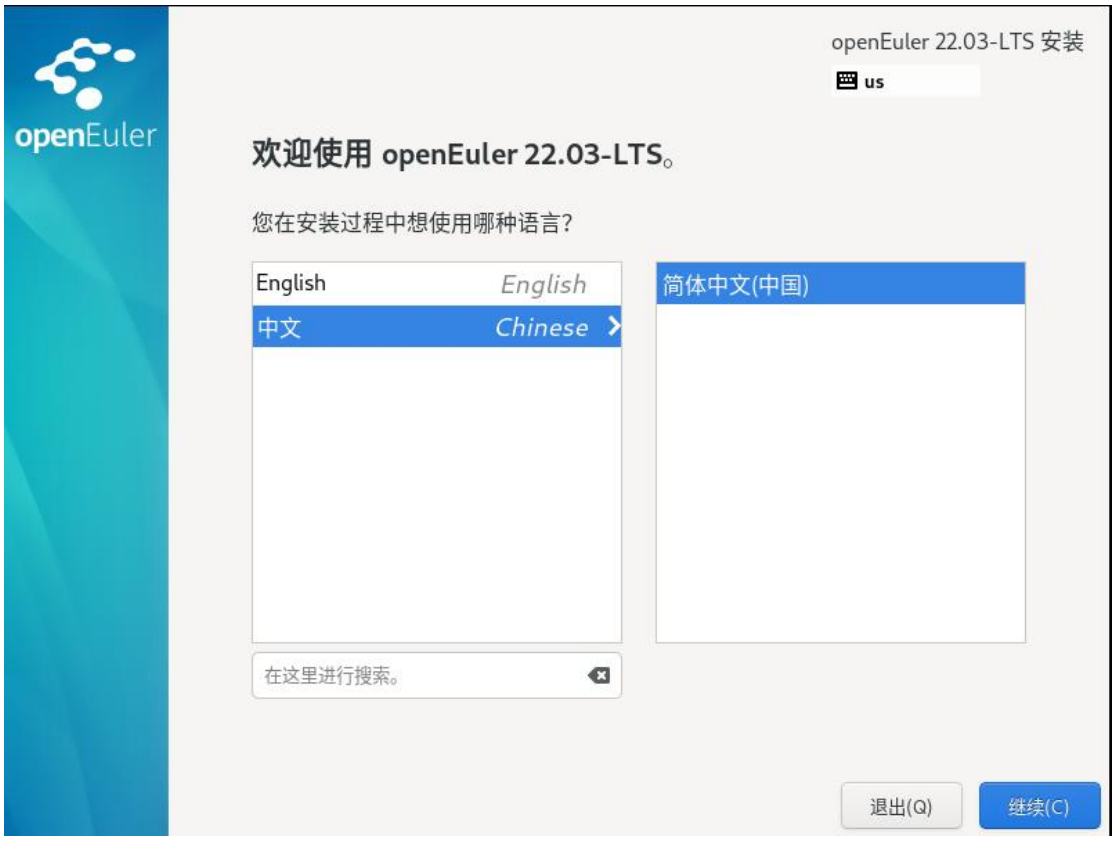
经过最后确认后开始创建虚拟机：



完成后打开虚拟机，进入安装流程：






openEuler
openEuler 22.03-LTS 安装
us
欢迎使用 openEuler 22.03-LTS。
您在安装过程中想使用哪种语言?
English English
中文 Chinese
简体中文(中国)
在这里进行搜索。
退出(Q)
继续(C)

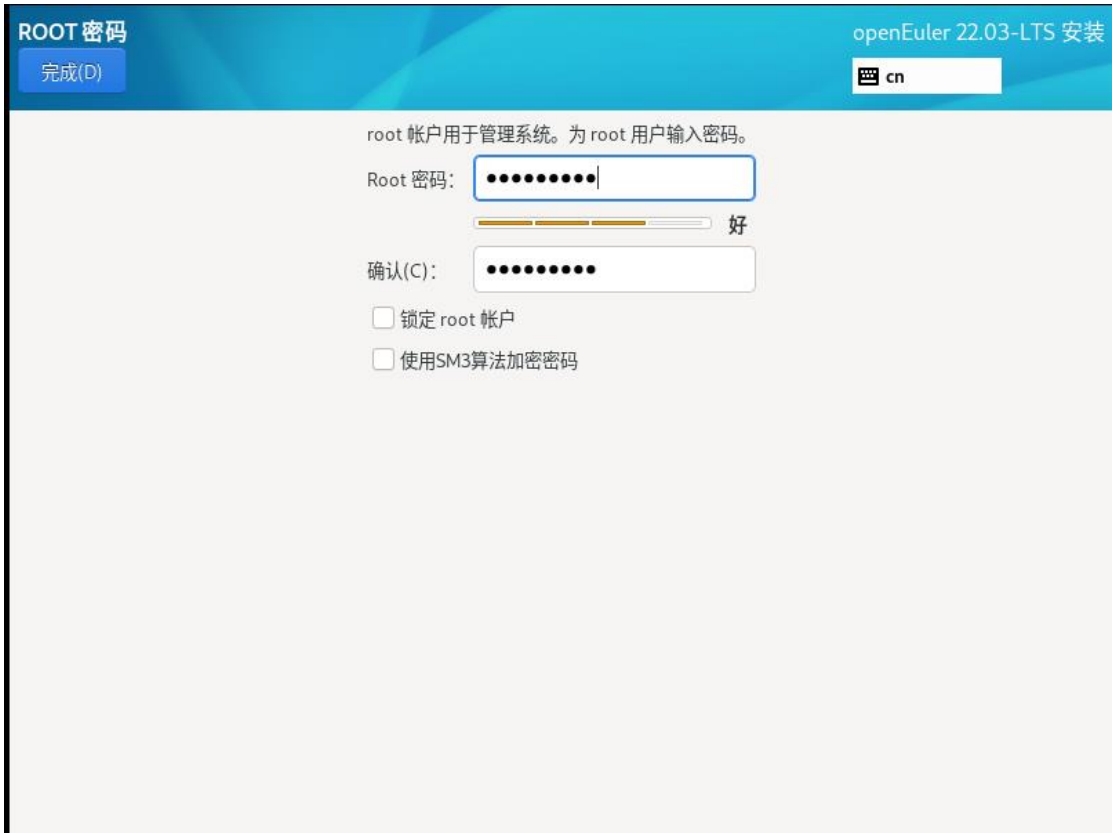

openEuler
安装信息摘要
openEuler 22.03-LTS 安装
cn
本地化
键盘(K)
汉语
语言支持 (L)
简体中文(中国)
时间和日期(T)
亚洲/上海 时区
软件
安装源(I)
LABEL=openEuler-22.03-LTS-x86_64:
软件选择 (S)
最小安装
系统
安装目的地(D)
已选择自动分区
网络和主机名(N)
未连接
用户设置
根密码(R)
root 账户已禁用。
退出(Q)
开始安装(B)
在点击“开始安装”按钮前我们并不会操作您的磁盘。
请先完成带有此图标标记的内容再进行下一步。

用户设置的 root 用户密码或新创建用户的密码均需要满足密码复杂度要求，否则会导致密码设置或用户创建失败。设置密码的复杂度的要求如下：

- 口令长度至少 8 个字符。
- 口令至少包含大写字母、小写字母、数字和特殊字符中的任意 3 种。
- 口令不能和账号一样。
- 口令不能使用字典词汇。

说明：在已装好的 openEuler 环境中，可以通过 cracklib-unpacker /usr/share/cracklib/pw_dict > dictionary.txt 命令导出字典库文件 dictionary.txt，用户可以查询密码是否在该字典中。

完成设置后，单击左上角的“完成”返回“安装概览”页面。



完成安装后重启系统，命令行提示 localhost login，这里输入之前安装过程中设置的用户名，或 root 用户，之后输入密码，需要提醒的是键入密码时不会显示包括已输入位在内的任何内容，要自行正确输入，完成后就可以通过控制台与 openEuler 交互了：

```
Authorized users only. All activities may be monitored and reported.
localhost login: root
Password:

Authorized users only. All activities may be monitored and reported.


Welcome to 5.10.0-60.18.0.50.oe2203.x86_64

System information as of time:  Thu Aug 24 11:57:58 PM CST 2023

System load:    0.69
Processes:      166
Memory used:    7.1%
Swap used:      0%
Usage On:       3%
IP address:     192.168.3.40
Users online:   1


[root@localhost ~]#
```

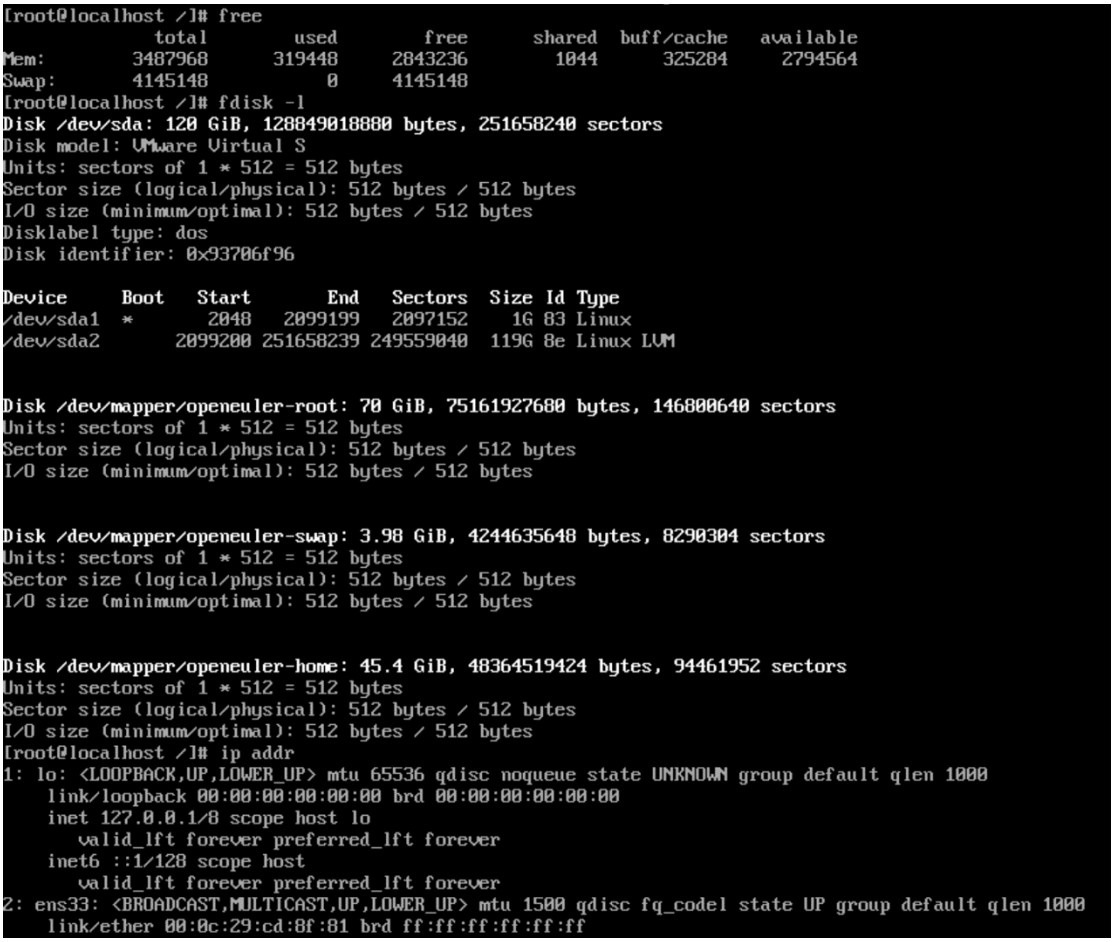
openEuler 使用的命令风格和 Linux 系统是一样的，具体内容可以参考 Linux 命令。

```
[root@localhost ~]# ls
anaconda-ks.cfg
[root@localhost ~]# cd ..
[root@localhost /]# ls
afs  boot  etc   lib    lost+found  mnt  proc  run   srv  tmp  var
bin  dev   home  lib64  media       opt  root  sbin  sys  usr
```

使用“cat /etc/os-release”命令查看系统信息，使用“lscpu”命令查看 CPU 信息：

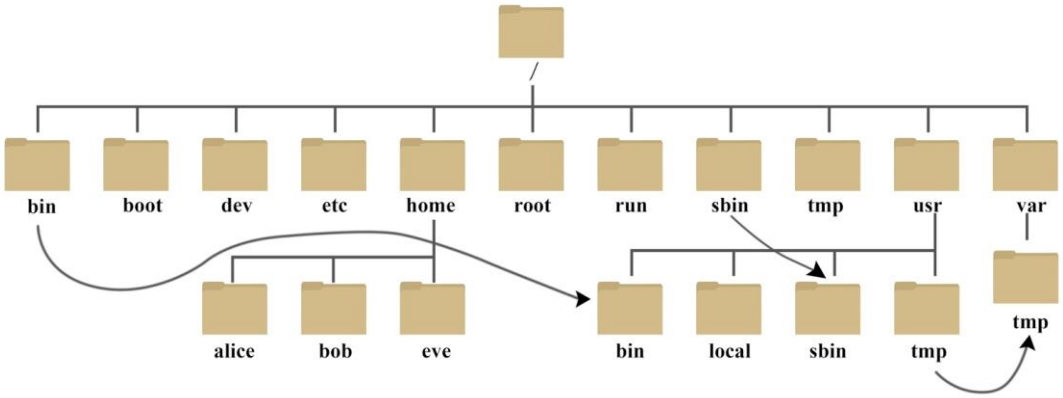
```
[root@localhost /]# cat /etc/os-release
NAME="openEuler"
VERSION="20.09"
ID="openEuler"
VERSION_ID="20.09"
PRETTY_NAME="openEuler 20.09"
ANSI_COLOR="0;31"

[root@localhost /]# lscpu
Architecture:                    x86_64
CPU op-mode(s):                  32-bit, 64-bit
Byte Order:                      Little Endian
Address sizes:                   43 bits physical, 48 bits virtual
CPU(s):                          4
On-line CPU(s) list:             0-3
Thread(s) per core:              1
Core(s) per socket:              1
Socket(s):                       4
NUMA node(s):                    1
Vendor ID:                       GenuineIntel
CPU family:                      6
Model:                           142
Model name:                      Intel(R) Core(TM) i7-7660U CPU @ 2.50GHz
Stepping:                        9
CPU MHz:                         2495.999
BogoMIPS:                        4991.99
Hypervisor vendor:               VMware
Virtualization type:             full
L1d cache:                       128 KiB
L1i cache:                       128 KiB
L2 cache:                        1 MiB
L3 cache:                        16 MiB
NUMA node0 CPU(s):               0-3
Vulnerability Itlb multihit:     KVM: Vulnerable
Vulnerability L1tf:              Mitigation; PTE Inversion
Vulnerability Mds:               Vulnerable: Clear CPU buffers attempted, no microcode; SMT Host state unknown
Vulnerability Meltdown:          Mitigation; PTI
Vulnerability Spec store bypass: Mitigation; Speculative Store Bypass disabled via prctl and seccomp
Vulnerability Spectre v1:        Mitigation; usercopy/swapgs barriers and __user pointer sanitization
Vulnerability Spectre v2:        Mitigation; Full generic retpoline, IBPB conditional, IBRS_FW, STIBP disabled, RSB filling
Vulnerability Srbds:             Unknown: Dependent on hypervisor status
Vulnerability Tsx async abort:   Vulnerable: Clear CPU buffers attempted, no microcode; SMT Host state unknown
Flags:                           fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ss syscall nx pdpe1gb rdtscp
                                 lm constant_tsc arch_perfmon nopl xtopology tsc_reliable nonstop_tsc cpuid pni pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 x2ap
                                 ic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand hypervisor lahf_lm abm 3dnowprefetch cpuid_fault invpcid_single pt
                                 i ssbd ibrs ibpb stibp fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 invpcid rtm mpx rdseed adx smap clflushopt xsaveopt xsavec x
                                 saves arat flush_l1d arch_capabilities
```

使用“free”命令查看内存信息，使用“fdisk -l”命令查看磁盘信息，使用“ip addr”命令查看 IP 地址：



3.Linux 系统目录结构与内核文件

本专题实验以 Linux 内核为核心，主要实验内容包括设计新的系统调用、动态模块、同步机制、设计改进进程调度算法、设计改进存贮管理方式、设计自己的设备驱动程序和添加新的文件系统等，目的在于通过实验掌握操作系统低层的实现机理，能根据实际运行环境构造自己的操作系统内核，因此需要经常和内核打交道，而要查阅内核的代码并对其作出修改，我们首先需要对 Linux 系统的文件目录结构有一定的了解：



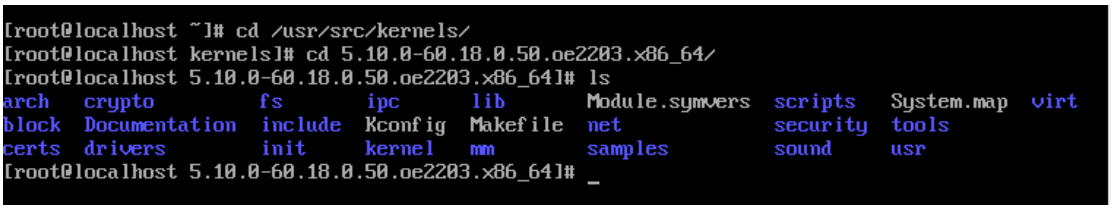
系统根目录下的文件目录名和对应功能如下表所示：

目录名	功能简介
/bin	bin 是 Binaries (二进制文件) 的缩写，这个目录存放着最经常使用的命令
/boot	这里存放的是启动 Linux 时使用的一些核心文件，包括一些连接文件以及镜像文件
/dev	dev 是 Device(设备) 的缩写，该目录下存放的是 Linux 的外部设备，在 Linux 中访问设备的方式和访问文件的方式是相同的
/etc	etc 是 Etcetera(等等) 的缩写,这个目录用来存放所有的系统管理所需要的配置文件和子目录
/home	用户的主目录，在 Linux 中，每个用户都有一个自己的目录，一般该目录名是以用户的账号命名的，如上图中的 alice、bob 和 eve

/lib	lib 是 Library(库) 的缩写这个目录里存放着系统最基本的动态连接共享库，其作用类似于 Windows 里的 DLL 文件。几乎所有的应用程序都需要用到这些共享库
/lost+found	这个目录一般情况下是空的，当系统非法关机后，这里就存放了一些文件
/media	linux 系统会自动识别一些设备，例如 U 盘、光驱等等，当识别后，Linux 会把识别的设备挂载到这个目录下
/mnt	系统提供该目录是为了让用户临时挂载别的文件系统的，我们可以将光驱挂载在 /mnt/ 上，然后进入该目录就可以查看光驱里的内容了
/opt	opt 是 optional(可选) 的缩写，这是给主机额外安装软件所摆放的目录。比如你安装一个 ORACLE 数据库则就可以放到这个目录下。默认是空的
/proc	proc 是 Processes(进程) 的缩写，/proc 是一种伪文件系统（也即虚拟文件系统），存储的是当前内核运行状态的一系列特殊文件，这个目录是一个虚拟的目录，它是系统内存的映射，我们可以通过直接访问这个目录来获取系统信息，这个目录的内容不在硬盘上而是在内存里，直接修改里面的某些文件可以起到修改系统配置的效果
/root	该目录为系统管理员，也称作超级权限者的用户主目录
/run	是一个临时文件系统，存储系统启动以来的信息

/sbin	s 就是 Super User 的意思，是 Superuser Binaries (超级用户的二进制文件) 的缩写，这里存放的是系统管理员使用的系统管理程序
/srv	该目录存放一些服务启动之后需要提取的数据
/sys	该目录下安装了一个文件系统 sysfs， sysfs 文件系统集成了下面 3 种文件系统的信息：针对进程信息的 proc 文件系统、针对设备的 devfs 文件系统以及针对伪终端的 devpts 文件系统。该文件系统是内核设备树的一个直观反映。当一个内核对象被创建的时候，对应的文件和目录也在内核对象子系统中被创建
/tmp	tmp 是 temporary(临时) 的缩写这个目录是用来存放一些临时文件的
/usr	**usr 是 unix shared resources(共享资源) 的缩写，这是一个非常重要的目录，用户的很多应用程序和文件都放在这个目录下**
/var	var 是 variable(变量) 的缩写，这个目录中存放着在不断扩充着的东西，我们习惯将那些经常被修改的目录放在这个目录下

通过 cd /usr/src/kernels 命令，进入存放内核相关内容的文件夹，接着通过 cd 5.10.0-60.18.0.50.oe2203.x86_64/ 命令，进入当前系统的内核文件夹，用 ls 命令查看文件夹里都有哪些文件：



目录/文件名	源码功能简介
/Documentation	说明文档，对每个目录的具体作用进行说明
/arch	不同 CPU 架构下的核心代码。其中的每一个子目录都代表 Linux 支持的 CPU 架构
/block	块设备通用函数
/certs	与证书相关
/crypto	常见的加密算法的 C 语言实现代码，譬如 crc32、md5、sha1 等
/drivers	内核中所有设备的驱动程序，其中的每一个子目录对应一种设备驱动
/include	内核编译通用的头文件
/init	内核初始化的核心代码
/ipc	内核中进程间的通信代码

/kernel	**内核的核心代码，此目录下实现了大多数 Linux 系统的内核函数。** **与处理器架构相关的内核代码在 /kernel/$ARCH/kernel**
/lib	内核共用的函数库 与处理器架构相关的库在/kernel/$ARCH/lib
/mm	内存管理代码，譬如页式存储管理内存的分配和释放等 与具体处理器架构相关的内存管理代码位于 /arch/$ARCH/mm 目录下
/net	网络通信相关代码
/samples	示例代码
/scripts	用于内核配置的脚本文件，用于实现内核配置的图形界面
/security	安全性相关的代码
/sound	与音频有关的代码，包括与音频有关的驱动程序
/tools	Linux 中的常用工具
/usr	该目录中的代码为内核尚未完全启动时执行用户空间代码提供了支持

/virt	此文件夹包含了虚拟化代码，它允许用户一次运行多个操作系统
COPYING	许可和授权信息
CREDITS	贡献者列表
Kbuild	内核设定脚本，可以对内核中的变量进行设定
Kconfig	配置哪些文件编译，哪些文件不用编译
Makefile	该文件将编译参数、编译所需的文件和必要的信息传给编译器

4.编辑器与编译器

要查阅内核的代码并对其作出修改再使之生效就需要我们掌握编辑器和编译器的使用，这里简单介绍 vi 编辑器和 gcc 编译器的使用。

4.1.vi 编辑器

所有的 Unix Like 系统都会内建 vi 文书编辑器，其他的文书编辑器则不一定会存在，目前使用比较多的是在 vi 基础上发展出的 vim 编辑器， vim 具有程序编辑的能力，可以主动的以字体颜色辨别语法的正确性，方便程序设计，操作上和 vi 基本一致。

vi/vim 共分为三种工作模式，分别是命令模式（Command mode），输入模式（Insert mode）和底线命令模式（Last line mode）。

这三种模式的作用分别是：

（1）命令模式：用户刚刚启动 vi，便进入了命令模式。

此状态下敲击键盘动作会被 vi 识别为命令，而非输入字符。比如我们此时按下 i，并不会输入一个字符，i 被当作了一个命令，以下是常用的几个命令：

- i 切换到输入模式，以输入字符。
- x 删除当前光标所在处的字符。
- ：切换到底线命令模式，以在最底一行输入命令。

若想要编辑文本：启动 Vi，进入了命令模式，按下 i，切换到输入模式。

（2）输入模式：在命令模式下按下 i 就进入了输入模式。

在输入模式中，可以使用以下按键：

- 字符按键以及 Shift 组合，输入字符
- ENTER，回车键，换行
- BACK SPACE，退格键，删除光标前一个字符
- DEL，删除键，删除光标后一个字符
- 方向键，在文本中移动光标
- HOME/END，移动光标到行首/行尾
- Page Up/Page Down，上/下翻页
- Insert，切换光标为输入/替换模式，光标将变成竖线/下划线
- ESC，退出输入模式，切换到命令模式

（3）底线命令模式：在命令模式下按下:（英文冒号）就进入了底线命令模式。

底线命令模式可以输入单个或多个字符的命令，可用的命令非常多，其中最基本的命令有（已经省略了冒号）：

- q 退出程序
- w 保存文件
- 按 ESC 键可随时退出底线命令模式。

4.2.gcc 编译器

gcc 与 g++ 分别是 gnu 的 c & c++ 编译器 gcc/g++ 在执行编译工作的时候，总共需要 4 步：

（1）预处理,生成 .i 的文件[预处理器 cpp]

（2）将预处理后的文件转换成汇编语言，生成文件 .s [编译器 egcs]

（3）有汇编变为目标代码(机器代码)生成 .o 的文件[汇编器 as]

（4）连接目标代码，生成可执行程序 [链接器 ld]

下表介绍了最常用的参数、作用以及使用样例：

参数	作用	样例
-c	只激活预处理、编译和汇编，把程序做成后缀名为.o 的 obj 文件	gcc -c hello.c
-s	只激活预处理和编译，把文件编译成为后缀名为.s 的汇编代码	gcc -S hello.c
-e	只激活预处理，不生成文件，需要把它重定向到一个输出文件里，会生成一个非常长的代码	gcc -E hello.c > test.txt gcc -E hello.c | less
-o	制定目标名称，默认的时候，gcc 编译出来的文件是 a.out	gcc -o hello.asm -S hello.c

-g	生成调试信息。GNU 调试器可利用该信息	
-I【大写 i】	指定额外的头文件搜索路径，有文件中包含如#include<file>时， gcc 会先在当前目录查找 file 文件，如果没有找到则回到默认的头文件目录 /usr/include 找；若使用 -I 指定了目录，则 gcc 会先在参数指定的目录查找，之后再按常规的顺序去找	gcc -o test test.c -I /usr/local/include/file
-L	指定额外的函数库搜索路径，使用方法同上	gcc -o test test.c -L /usr/test
-l【小写 L】	l 紧跟库名（如数学库的全名是 libm.so，去掉 lib 前缀和.so 后缀得到库名为 m），库文件默认在 /usr/lib 目录下搜索	gcc -o test test.c -lm

4.3.用管道命令 less 在终端中翻页

我们翻阅目录和打开文件后，经常会由于输出内容太多，使一屏无法显示，为了查阅我们想看到的内容，就需要在输入命令的后面加上管道命令（用编程时常用的或逻辑号'|' 标识）：less 或者 more，具体的使用方法请自行在网络上搜索，这里只简单介绍 less 的使用方法，它和 vi 非常类似：

下面是一个样例：

```
[root@localhost Documentation]# date --help_
```

输出结果如下：

```
  %y   last two digits of year (00..99)
  %Y   year
  %z   +hhmm numeric time zone (e.g., -0400)
  %:z  +hh:mm numeric time zone (e.g., -04:00)
  %::z  +hh:mm:ss numeric time zone (e.g., -04:00:00)
  %:::z  numeric time zone with : to necessary precision (e.g., -04, +05:30)
  %Z   alphabetic time zone abbreviation (e.g., EDT)

By default, date pads numeric fields with zeroes.
The following optional flags may follow '%':

  -  (hyphen) do not pad the field
  _  (underscore) pad with spaces
  0  (zero) pad with zeros
  +  pad with zeros, and put '+' before future years with >4 digits
  ^  use upper case if possible
  #  use opposite case if possible

After any flags comes an optional field width, as a decimal number;
then an optional modifier, which is either
E to use the locale's alternate representations if available, or
O to use the locale's alternate numeric symbols if available.

Examples:
Convert seconds since the epoch (1970-01-01 UTC) to a date
  $ date --date='@2147483647'

Show the time on the west coast of the US (use tzselect(1) to find TZ)
  $ TZ='America/Los_Angeles' date

Show the local time for 9AM next Friday on the west coast of the US
  $ date --date='TZ="America/Los_Angeles" 09:00 next Fri'

GNU coreutils online help: <https://www.gnu.org/software/coreutils/>
Full documentation <https://www.gnu.org/software/coreutils/date>
or available locally via: info '(coreutils) date invocation'
[root@localhost Documentation]#
```

相关内容的前面很多行无法看到，接下来我们使用 less（-N 参数用于显示行号）：

```
[root@localhost Documentation]# date --help|less -N
```

下面是输出结果，这是一个类似于 vi 的，可以交互的界面：

less 的动作命令如下:

- j 向下移动一行；同 vi
- k 向上移动一行；同 vi
- f 向下滚动一屏；forword
- b 向上滚动一屏；backword
- head -n 10 /etc/profile 显示/etc/profile 的前 10 行内容
- tail -n 5 /etc/profile 显示/etc/profile 的最后 5 行内容

5.将文件上传到虚拟机

设置静态 IP 地址，本例使用了 192.168.152.200/24：

```
[root@localhost vsftpd]# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:cd:8f:81 brd ff:ff:ff:ff:ff:ff
    inet 192.168.152.129/24 brd 192.168.152.255 scope global dynamic noprefixroute ens33
       valid_lft 1720sec preferred_lft 1720sec
    inet6 fe80::77de:7049:155d:13e2/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[root@localhost vsftpd]# ip addr show dev ens33
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:cd:8f:81 brd ff:ff:ff:ff:ff:ff
    inet 192.168.152.129/24 brd 192.168.152.255 scope global dynamic noprefixroute ens33
       valid_lft 1488sec preferred_lft 1488sec
    inet6 fe80::77de:7049:155d:13e2/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
[root@localhost vsftpd]# ip address add 192.168.152.200/24 dev ens33
[root@localhost vsftpd]# ip addr show dev ens33
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:cd:8f:81 brd ff:ff:ff:ff:ff:ff
    inet 192.168.152.129/24 brd 192.168.152.255 scope global dynamic noprefixroute ens33
       valid_lft 1394sec preferred_lft 1394sec
    inet 192.168.152.200/24 scope global secondary ens33
       valid_lft forever preferred_lft forever
    inet6 fe80::77de:7049:155d:13e2/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

使用 vsftpd 需要先安装 vsftpd 软件，在已经配置 yum 源的情况下，通过 root 权限执行“dnf install vsftpd”命令，即可完成 vsftpd 的安装。接着执行“dnf install net-tools”命令，安装 net-tools 包使得 netstat 命令可可用，通过该命令查看通信端口 21 是否开启，如下显示说明 vsftpd 已经启动：

```
[root@localhost vsftpd]# netstat -tulnp | grep 21
tcp6       0      0 :::21                   :::*                    LISTEN      3717/vsftpd
```

需要进行其它管理操作时，命令“systemctl stop vsftpd”和“systemctl restart vsftpd”分别可以停止和重启 vsftpd 服务，更细致

的权限管理可以通过修改 vsftpd 的配置文件实现，配置文件的路径可以参考下表：

配置文件	含义
/etc/vsftpd/vsftpd.conf	vsftpd 进程的主配置文件，配置内容格式为“参数=参数值”，且参数和参数值不能为空。vsftpd.conf 的详细介绍可以使用如下命令查看： man 5 vsftpd.conf
/etc/pam.d/vsftpd	PAM（Pluggable Authentication Modules）认证文件，主要用于身份认证和限制一些用户的操作。
/etc/vsftpd/ftpusers	禁用使用 vsftpd 的用户列表文件。默认情况下，系统帐号也在该文件中，因此系统帐号默认无法使用 vsftpd。
/etc/vsftpd/user_list	禁止或允许登录 vsftpd 服务器的用户列表文件。该文件是否生效，取决于主配置文件 vsftpd.conf 中的如下参数： userlist_enable：是否启用 userlist 机制，YES 为启用，此时 userlist_deny 配置有效，NO 为禁用。

	userlist_deny：是否禁止 user_list 中的用户登录，YES 为禁止名单中的用户登录，NO 为允许命令中的用户登录。 例如 userlist_enable=YES，userlist_deny=YES，则 user_list 中的用户都无法登录。
/etc/vsftpd/chroot_list	是否限制在主目录下的用户列表。该文件默认不存在，需要手动建立。它是主配置文件 vsftpd.conf 中参数 chroot_list_file 的参数值。 其作用是限制还是允许，取决于主配置文件 vsftpd.conf 中的如下参数： •chroot_local_user：是否将所有用户限制在主目录，YES 为启用，NO 禁用。 •chroot_list_enable：是否启用限制用户的名单，YES 为启用，NO 禁用。 例如 chroot_local_user=YES，chroot_list_enable=YES，且指定 chroot_list_file=/etc/vsftpd/chroot_list 时，表示所有用户被限制在其主目

	录下，而 chroot_list 中的用户不受限制。
/usr/sbin/vsftpd	vsftpd 的唯一执行文件。
/var/ftp/	匿名用户登录的默认根目录，与 ftp 帐户的用户主目录有关。

下图是主配置文件的默认状态：

```
# Example config file /etc/vsftpd/vsftpd.conf
#
# The default compiled in settings are fairly paranoid. This sample file
# loosens things up a bit, to make the ftp daemon more usable.
# Please see vsftpd.conf.5 for all compiled in defaults.
#
# READ THIS: This example file is NOT an exhaustive list of vsftpd options.
# Please read the vsftpd.conf.5 manual page to get a full idea of vsftpd's
# capabilities.
#
# Allow anonymous FTP? (Beware - allowed by default if you comment this out).
anonymous_enable=NO
#
# Uncomment this to allow local users to log in.
# When SELinux is enforcing check for SE bool ftp_home_dir
local_enable=YES
#
# Uncomment this to enable any form of FTP write command.
write_enable=YES
#
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftpd's)
local_umask=022
#
# Uncomment this to allow the anonymous FTP user to upload files. This only
# has an effect if the above global write enable is activated. Also, you will
# obviously need to create a directory writable by the FTP user.
# When SELinux is enforcing check for SE bool allow_ftpd_anon_write, allow_ftpd_full_access
#anon_upload_enable=YES
#
# Uncomment this if you want the anonymous FTP user to be able to create
# new directories.
#anon_mkdir_write_enable=YES
#
# Activate directory messages - messages given to remote users when they
# go into a certain directory.
"vsftpd.conf" 127L, 5098C
```

```
# Activate directory messages - messages given to remote users when they
# go into a certain directory.
dirmessage_enable=YES
#
# Activate logging of uploads/downloads.
xferlog_enable=YES
#
# Make sure PORT transfer connections originate from port 20 (ftp-data).
connect_from_port_20=YES
#
# If you want, you can arrange for uploaded anonymous files to be owned by
# a different user. Note! Using "root" for uploaded files is not
# recommended!
#chown_uploads=YES
#chown_username=whoever
#
# You may override where the log file goes if you like. The default is shown
# below.
#xferlog_file=/var/log/xferlog
#
# If you want, you can have your log file in standard ftpd xferlog format.
# Note that the default log file location is /var/log/xferlog in this case.
xferlog_std_format=YES
#
# You may change the default value for timing out an idle session.
#idle_session_timeout=600
#
# You may change the default value for timing out a data connection.
#data_connection_timeout=120
#
# It is recommended that you define on your system a unique user which the
# ftp server can use as a totally isolated and unprivileged user.
#nopriv_user=ftpsecure
#
# Enable this and the server will recognise asynchronous ABOR requests. Not
# recommended for security (the code is non-trivial). Not enabling it,
```

```
#async_abor_enable=YES
#
# By default the server will pretend to allow ASCII mode but in fact ignore
# the request. Turn on the below options to have the server actually do ASCII
# mangling on files when in ASCII mode. The vsftpd.conf(5) man page explains
# the behaviour when these options are disabled.
# Beware that on some FTP servers, ASCII support allows a denial of service
# attack (DoS) via the command "SIZE /big/file" in ASCII mode. vsftpd
# predicted this attack and has always been safe, reporting the size of the
# raw file.
# ASCII mangling is a horrible feature of the protocol.
#ascii_upload_enable=YES
#ascii_download_enable=YES
#
# You may fully customise the login banner string:
#ftpd_banner=Welcome to blah FTP service.
#
# You may specify a file of disallowed anonymous e-mail addresses. Apparently
# useful for combatting certain DoS attacks.
#deny_email_enable=YES
# (default follows)
#banned_email_file=/etc/vsftpd/banned_emails
#
# You may specify an explicit list of local users to chroot() to their home
# directory. If chroot_local_user is YES, then this list becomes a list of
# users to NOT chroot().
# (Warning! chroot'ing can be very dangerous. If using chroot, make sure that
# the user does not have write access to the top level directory within the
# chroot)
#chroot_local_user=YES
#chroot_list_enable=YES
# (default follows)
#chroot_list_file=/etc/vsftpd/chroot_list
#
# You may activate the "-R" option to the builtin ls. This is disabled by
# default to avoid remote users being able to cause excessive I/O on large
```

```
# sites. However, some broken FTP clients such as "ncftp" and "mirror" assume
# the presence of the "-R" option, so there is a strong case for enabling it.
#ls_recurse_enable=YES
#
# When "listen" directive is enabled, vsftpd runs in standalone mode and
# listens on IPv4 sockets. This directive cannot be used in conjunction
# with the listen_ipv6 directive.
listen=NO
#
# This directive enables listening on IPv6 sockets. By default, listening
# on the IPv6 "any" address (::) will accept connections from both IPv6
# and IPv4 clients. It is not necessary to listen on *both* IPv4 and IPv6
# sockets. If you want that (perhaps because you want to listen on specific
# addresses) then you must run two copies of vsftpd with two configuration
# files.
# Make sure, that one of the listen options is commented !!
listen_ipv6=YES

pam_service_name=vsftpd
userlist_enable=YES
```

包含的参数和含义如下表所示：

参数	含义
anonymous_enable	是否允许匿名用户登录，YES 为允许匿名登录，NO 为不允许。
local_enable	是否允许本地用户登入，YES 为允许本地用户登入，NO 为不允许。
write_enable	是否允许登录用户有写权限，YES 为启用上传写入功能，NO 为禁用。
local_umask	本地用户新增档案时的 umask 值。
dirmessage_enable	当用户进入某个目录时，是否显示该目录需要注意的内容，YES 为显示注意内容，NO 为不显示。
xferlog_enable	是否记录使用者上传与下载文件的操作，YES 为记录操作，NO 为不记录。
connect_from_port_20	Port 模式进行数据传输是否使用端口 20，YES 为使用端口 20，NO 为不使用端口 20。
xferlog_std_format	传输日志文件是否以标准 xferlog 格式书写，YES 为使用该格式书写，NO 为不使用。
listen	设置 vsftpd 是否以 stand alone 的方式启动，YES 为使用 stand alone 方式启动，NO 为不使用该方式。

pam_service_name	支持 PAM 模块的管理，配置值为服务名称，例如 vsftpd。
userlist_enable	是否支持/etc/vsftpd/user_list 文件内的账号登录控制，YES 为支持，NO 为不支持。
tcp_wrappers	是否支持 TCP Wrappers 的防火墙机制，YES 为支持，NO 为不支持。
listen_ipv6	是否侦听 IPv6 的 FTP 请求，YES 为侦听，NO 为不侦听。listen 和 listen_ipv6 不能同时开启。

openEuler 系统中，vsftpd 默认使用 GMT 时间（格林尼治时间），可能和本地时间不一致，为了避免服务器和客户端时间不一致，在上传下载文件时可能引起错误，我们需要将 vsftpd 使用的时间改为本地时间，首先用 vi 打开 vsftpd 的主配置文件，路径为“/etc/vsftpd/vsftpd.conf”，接着在该文件末尾加上一句配置内容“use_localtime=YES”，同时加入欢迎信息配置“banner_file=/etc/vsftpd/welcome.txt”，保存后重启 vsftpd 并将该服务设置为开机启动：

```
[root@localhost vsftpd]# systemctl restart vsftpd
[root@localhost vsftpd]# systemctl enable vsftpd
Created symlink /etc/systemd/system/multi-user.target.wants/vsftpd.service → /usr/lib/systemd/system
/vsftpd.service.
```

之后用 vi 建立 welcome.txt 文件，并写入希望的欢迎语。

因为在虚拟机上运行，安全性不需要做过多考虑，直接开放匿名用户的读写权限，我们将部分参数修改如下：

anonymous_enable=YES

anon_umask=022

anon_upload_enable=YES

anon_mkdir_write_enable=YES

anon_other_write_enable=YES

通过“useradd visitor”命令添加新用户 visitor，通过“passwd visitor”命令修改该用户的密码，本例设定为“123”；我们用这个新用户验证是否配置成功，openEuler 系统精简版没有 ftp 命令，可以在 root 权限下执行 dnf install ftp 命令安装后再使用 ftp 命令，输入“ftp localhost”命令，用户名为 visitor，密码为 123，可以得到结果如下：

也可以使用匿名用户，用户名为 anonymous，密码为空，结果如下：

如果要将 FTP 开放给 Internet 使用，需要在 root 权限下对防火墙和 SElinux 进行设置：

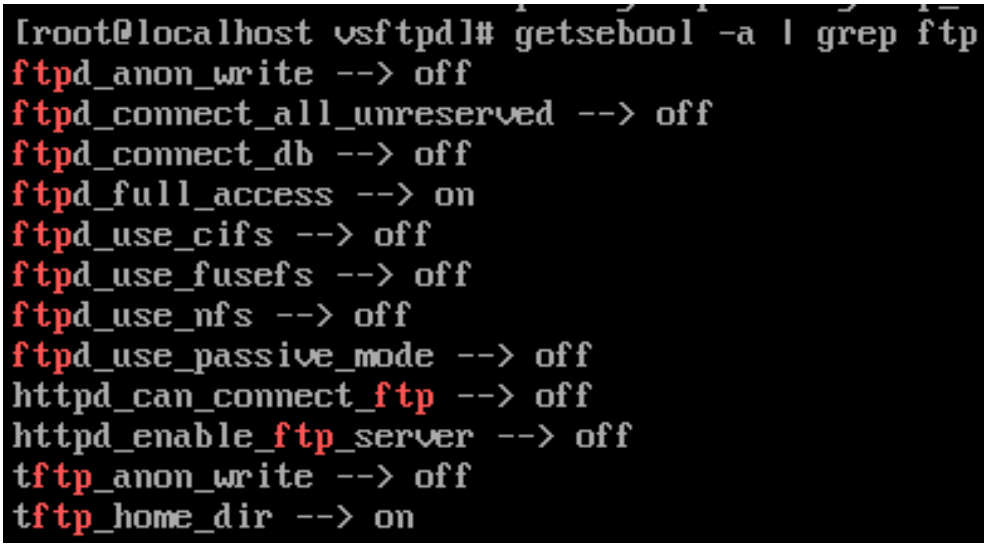
firewall-cmd --add-service=ftp --permanent //永久开放 ftp 服务

firewall-cmd --reload //重新载入防火墙配置

chown ftp /var/ftp/pub //设置 pub/文件夹的所有者为 ftp

setsebool -P ftpd_full_access on //对 SELinux 的限制作出修改

我们可以用“getsebool -a | grep ftp”命令进行检查，修改后得到结果如下：



此时我们在同一局域网内的计算机上在文件管理器的地址栏中输入 ftp://192.168.152.200 就可以直接访问 ftp 服务器了并操作其中的文件了。特别的，ftp 服务器存储文件的目录在系统的“var/ftp/pub/”位置。