**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**

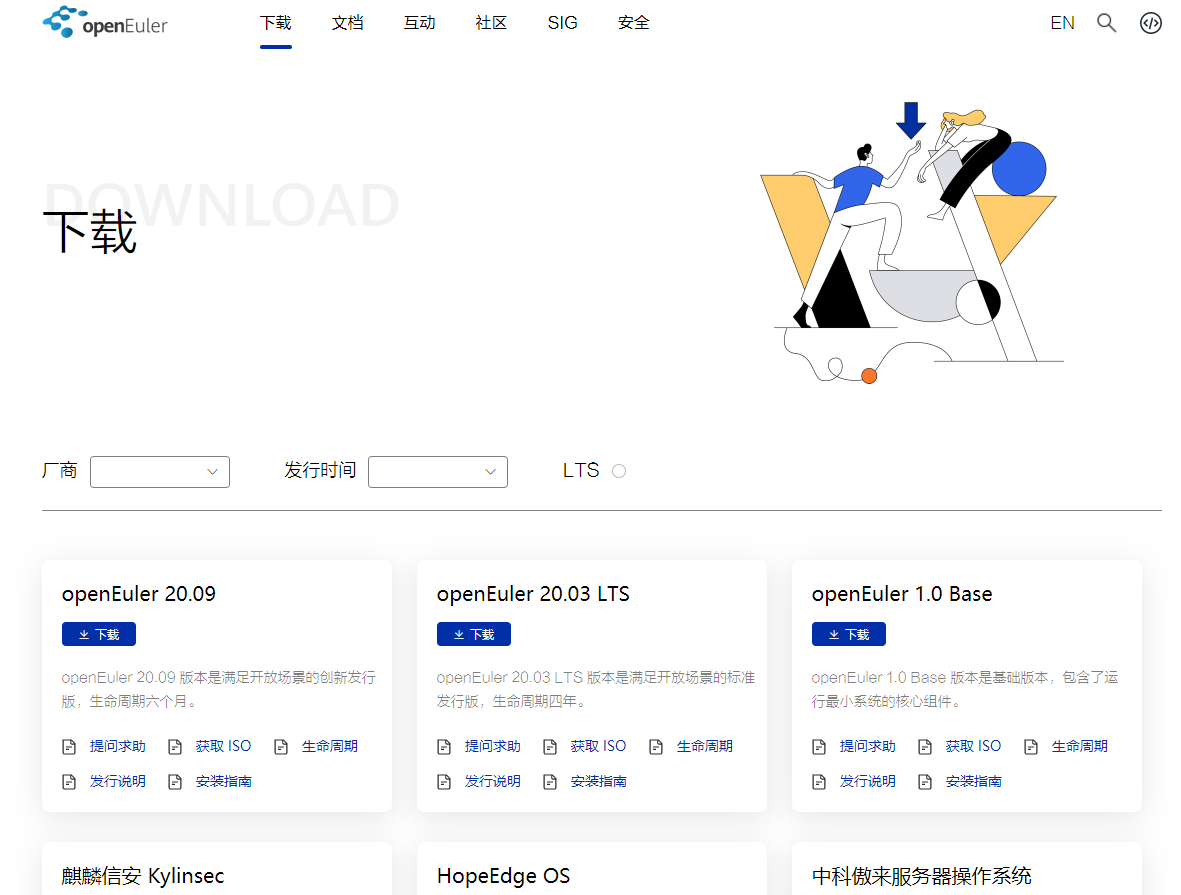
openEuler是一款开源操作系统。当前openEuler内核源于Linux，支持鲲鹏及其它多种处理器，能够充分释放计算芯片的潜能，是由全球开源贡献者构建的高效、稳定、安全的开源操作系统，适用于数据库、大数据、云计算、人工智能等应用场景。同时，openEuler是一个面向全球的操作系统开源社区，通过社区合作，打造创新平台，构建支持多处理器架构、统一和开放的操作系统，推动软硬件应用生态繁荣发展。

**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”



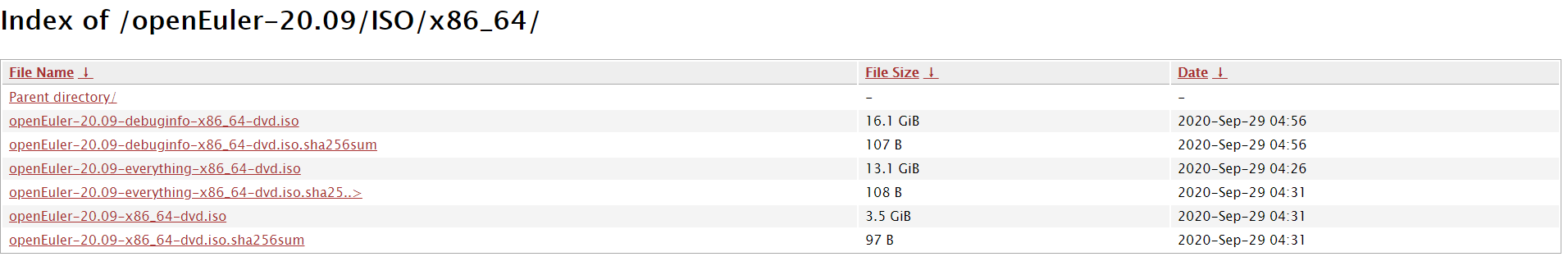
点击openEurer 20.09版本下“获取ISO”链接，进入下载目录：



进入下载目录后有三个文件夹，其中“aarch64”文件夹里放的是AArch64架构下openEuler的ISO和虚拟机镜像，“source”文件夹里放的是openEuler的源码ISO，“x86\_64”文件夹里放的是x86\_64架构下openEuler的ISO和虚拟机镜像：



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** |
| openEuler-20.09-x86\_64-dvd.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 |
| openEuler-20.09-everything-x86\_64-dvd.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 |
| openEuler-20.09-debuginfo-x86\_64-dvd.iso | x86\_64架构下openEuler的调试ISO，包含了调试所需的符号表信息 |
| \*.sha256sum | 用于校验ISO文件版本和完整性的文件 |

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。环境配置的过程中我们不进行这一步骤。

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 4.19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：



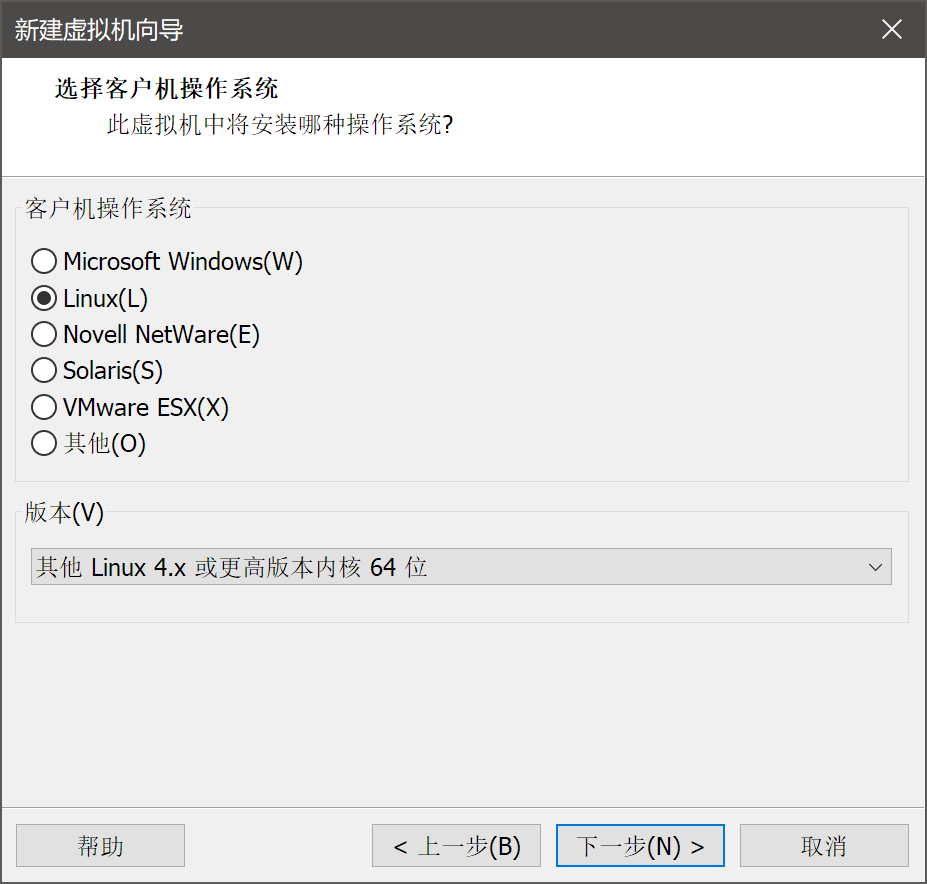
兼容性方面不需要进行设置，直接进入下一步：



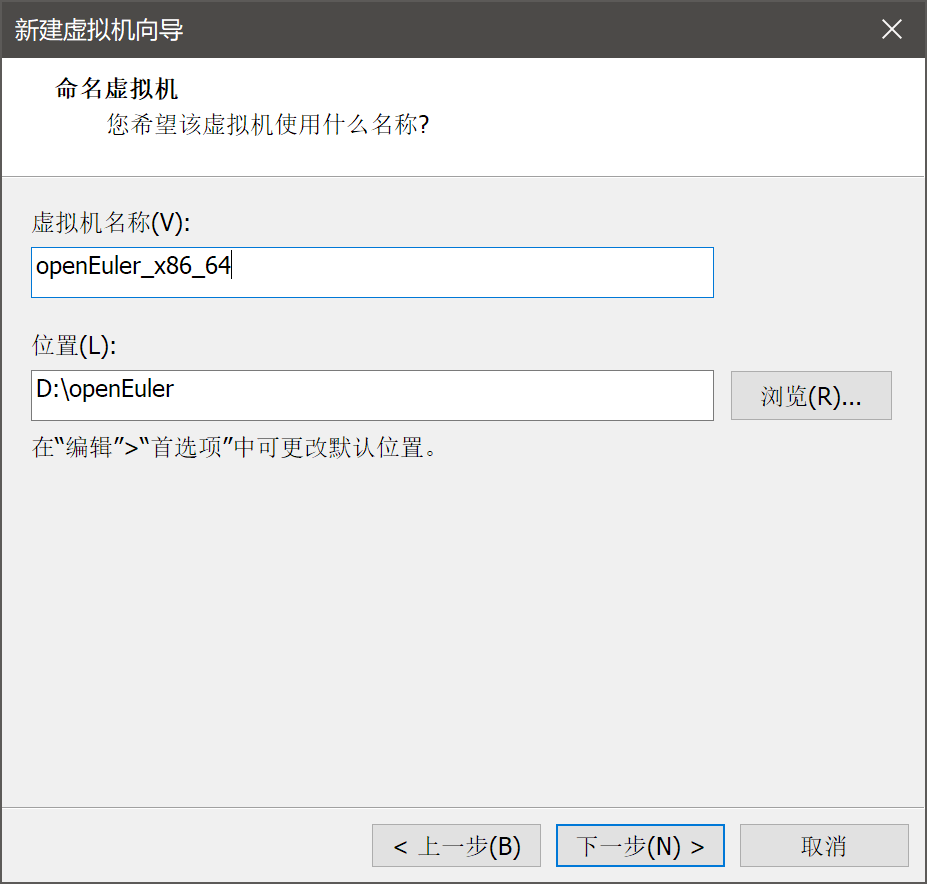
选择ISO文件所在的文件目录，这里的“无法检测”正是因为VMware没有提供自动化的配置选项，不会影响配置流程，直接进入下一步：



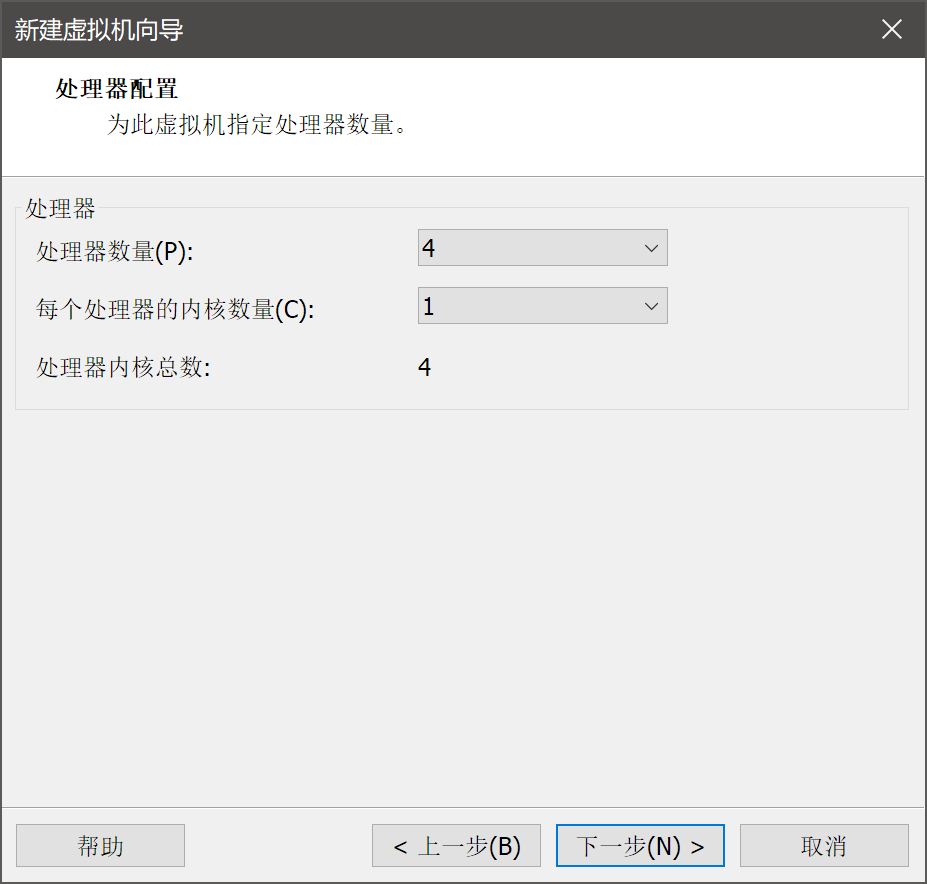
openEuler系统的设计基于Linux 4.19版本的内核，因此选择“其它 Linux 4.x 或更高版本内核64位”：



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：



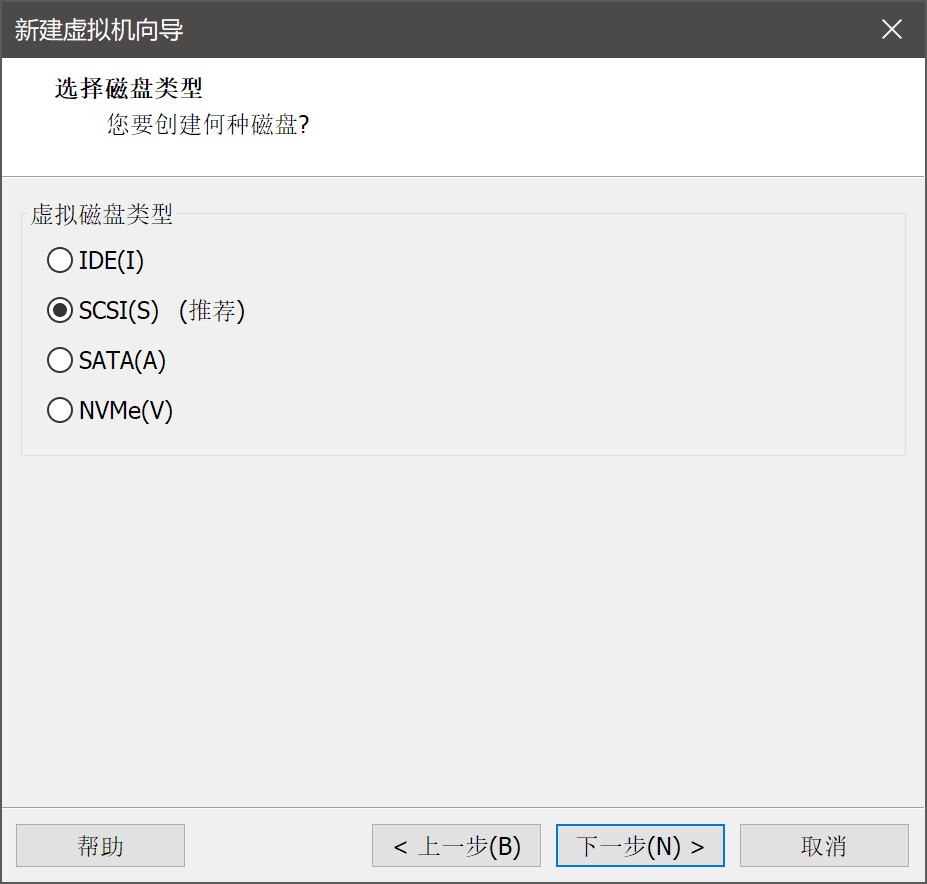
对虚拟机的网络配置进行设置：



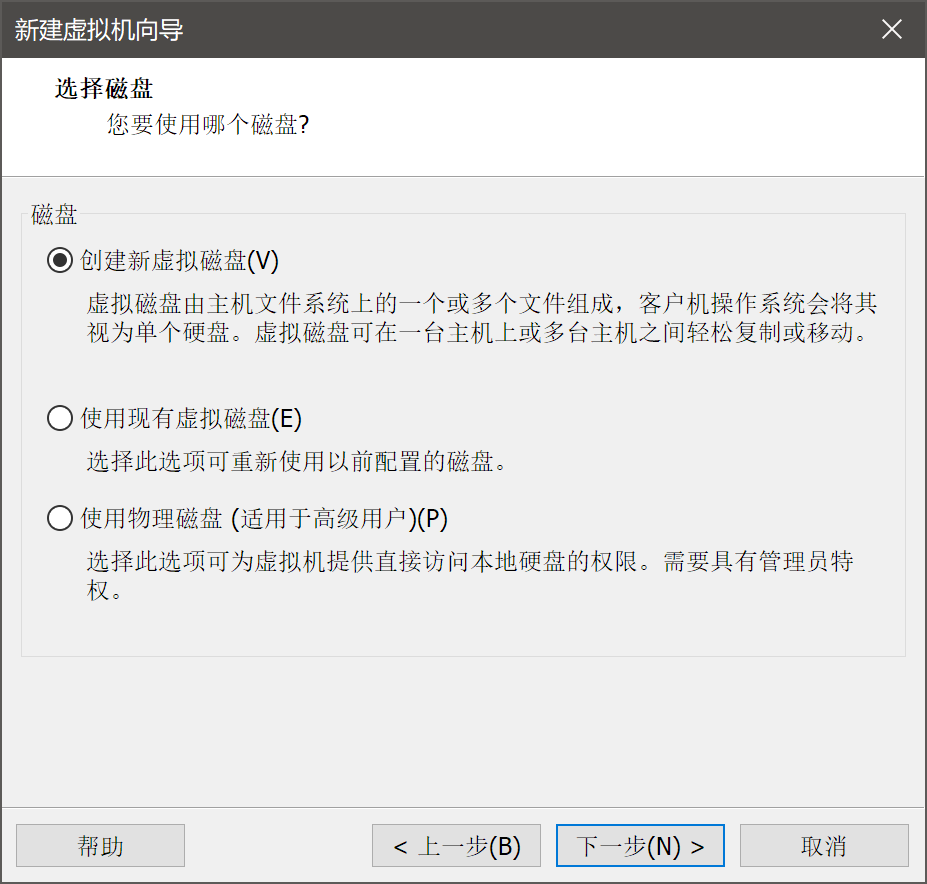
对虚拟机的 I/O 进行设置：



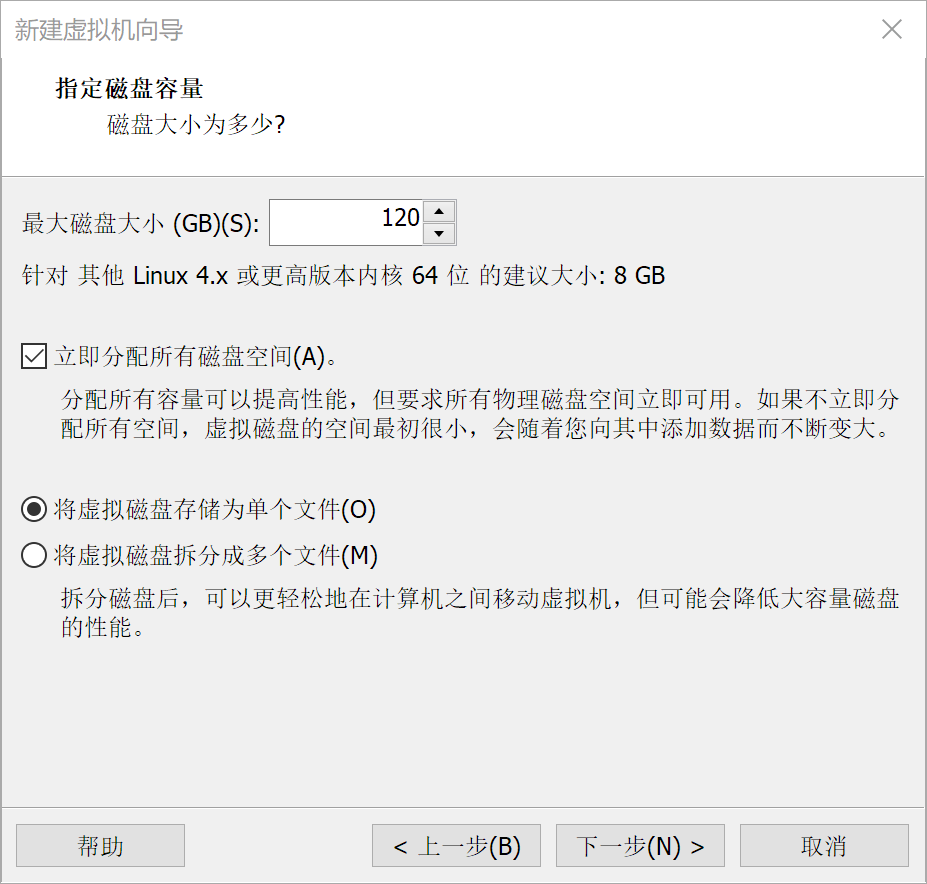
对虚拟机所使用的“外存（磁盘）”进行设置：



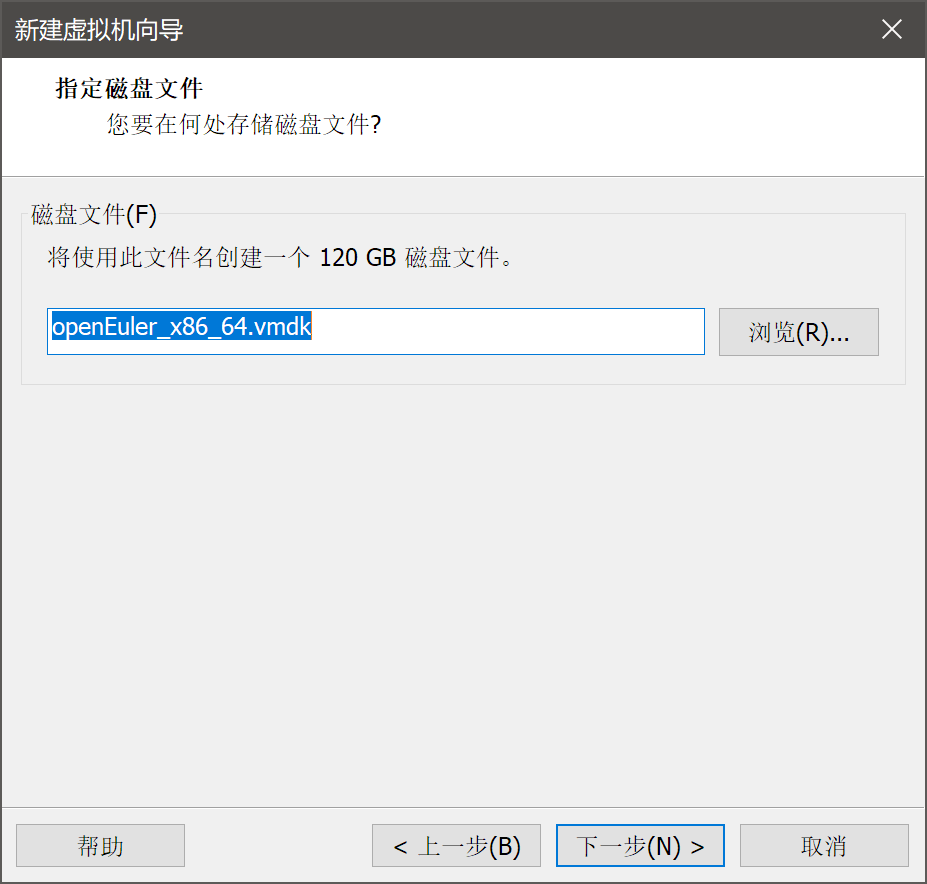
为虚拟机使用的“外存”分配存储空间：



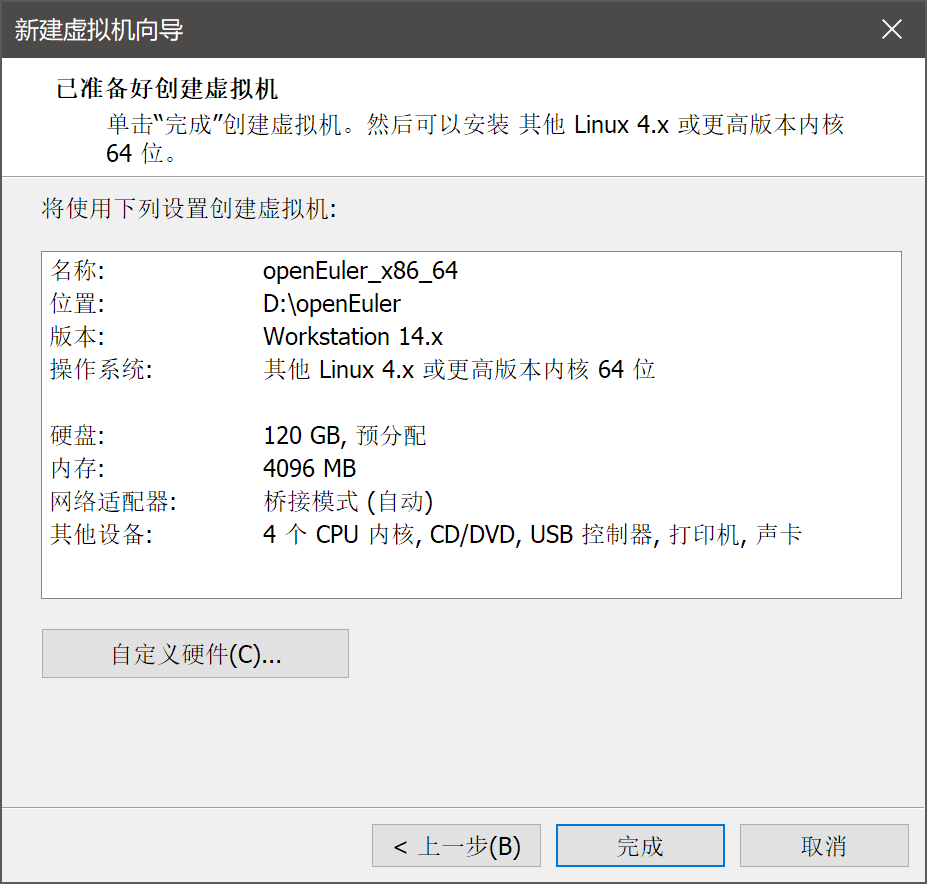
根据官网文档分配120G磁盘空间给虚拟机，为读写性能考虑选择将虚拟磁盘存储为单个文件：



选择该虚拟磁盘文件的存储路径：

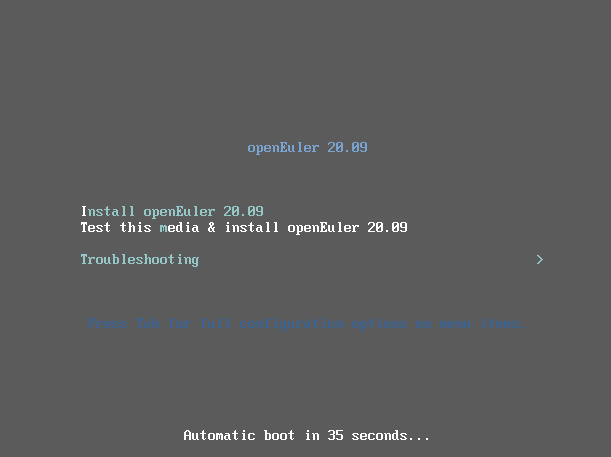


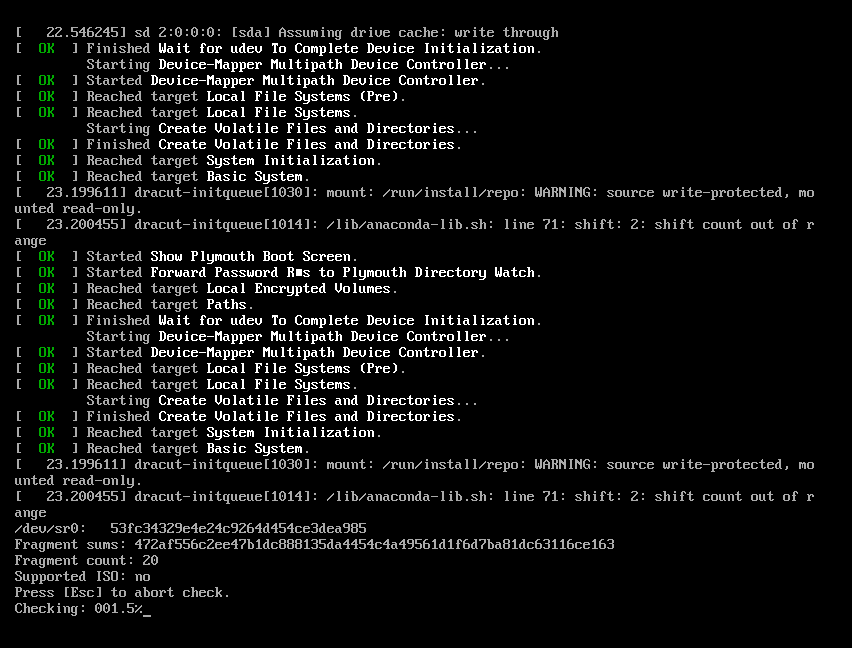
经过最后确认后开始创建虚拟机：

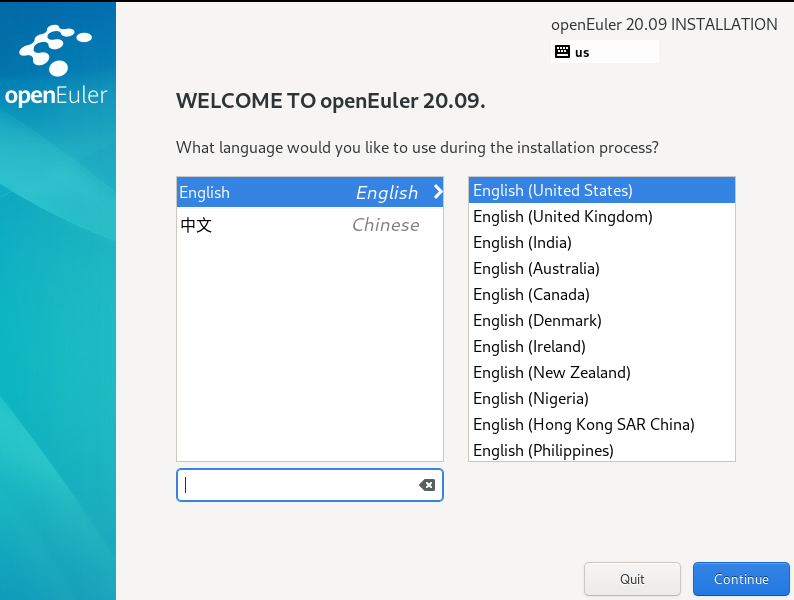




完成后打开虚拟机，进入安装流程：









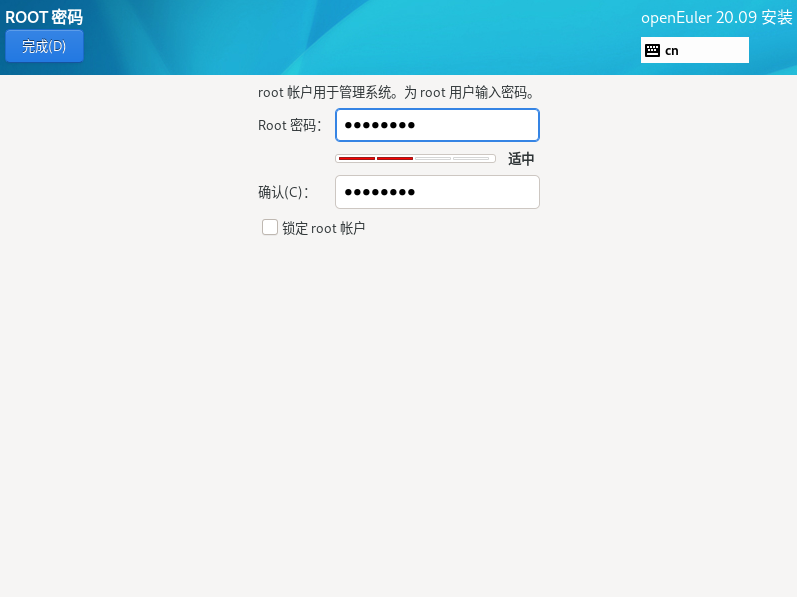


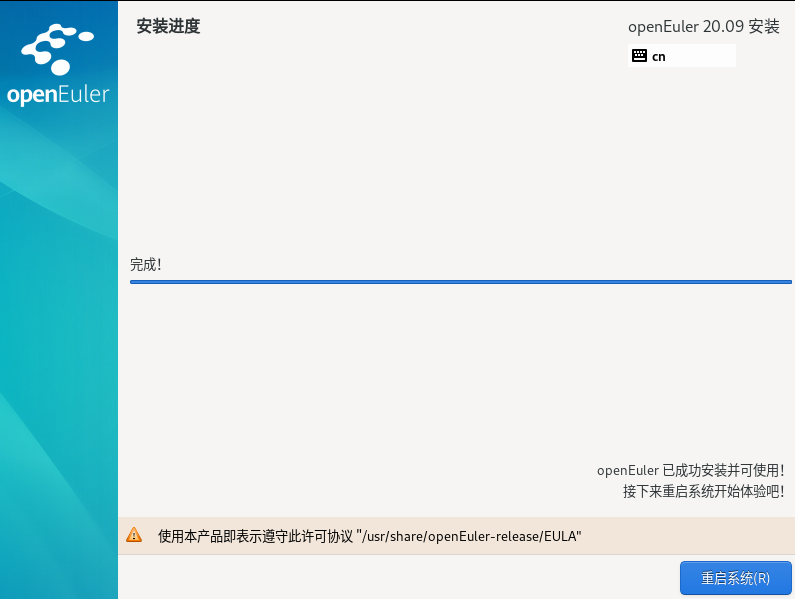
用户设置的root用户密码或新创建用户的密码均需要满足密码复杂度要求，否则会导致密码设置或用户创建失败。设置密码的复杂度的要求如下：

* 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。

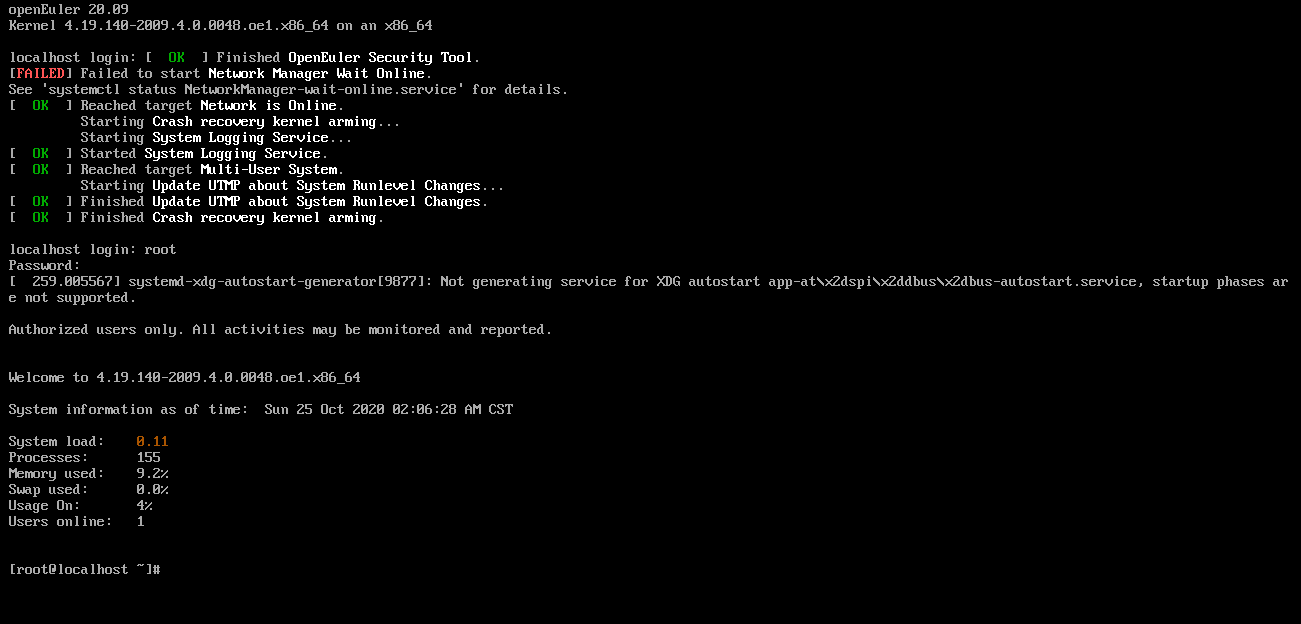
**说明：**在已装好的openEuler环境中，可以通过cracklib-unpacker /usr/share/cracklib/pw\_dict > dictionary.txt命令导出字典库文件dictionary.txt，用户可以查询密码是否在该字典中。

完成设置后，单击左上角的“完成”返回“安装概览”页面。

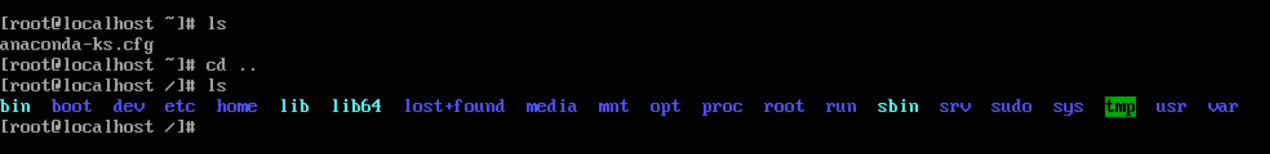




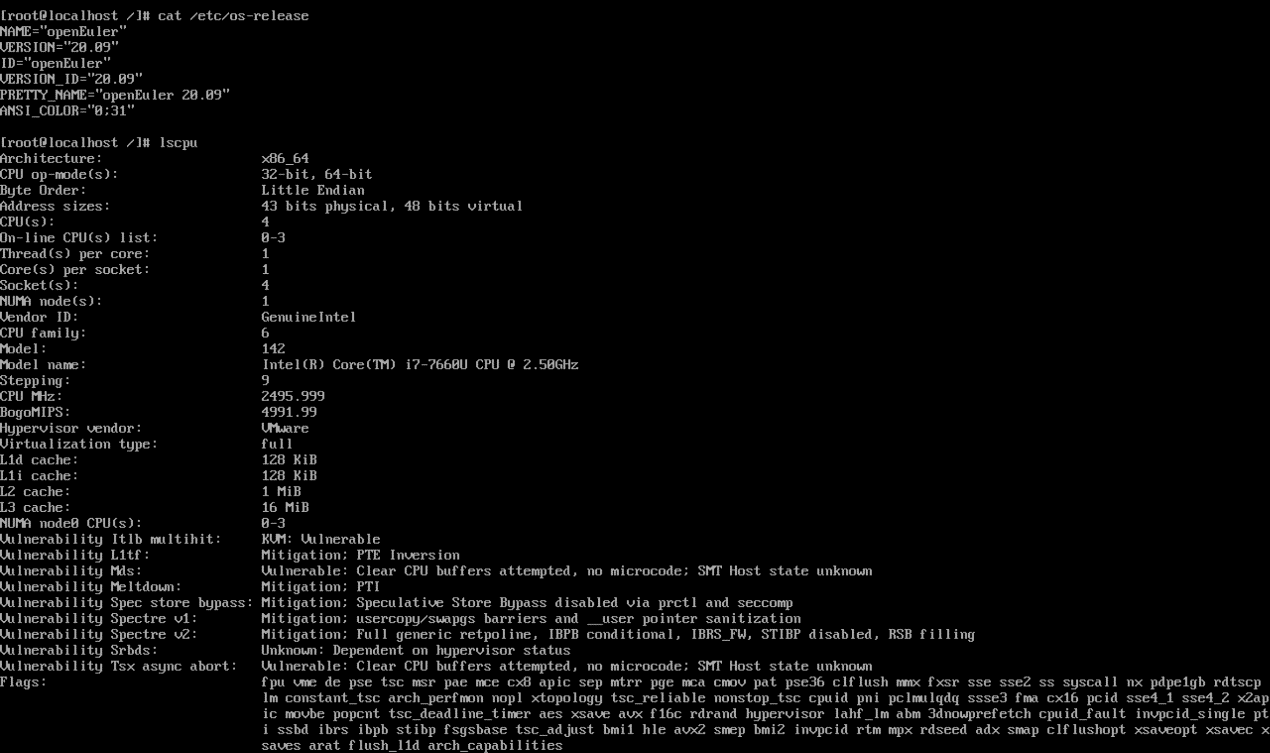
完成安装后重启系统，命令行提示localhost login，这里输入之前安装过程中设置的用户名，或root用户，之后输入密码，需要提醒的是键入密码时不会显示包括已输入位在内的任何内容，要自行正确输入，完成后就可以通过控制台与openEuler交互了：



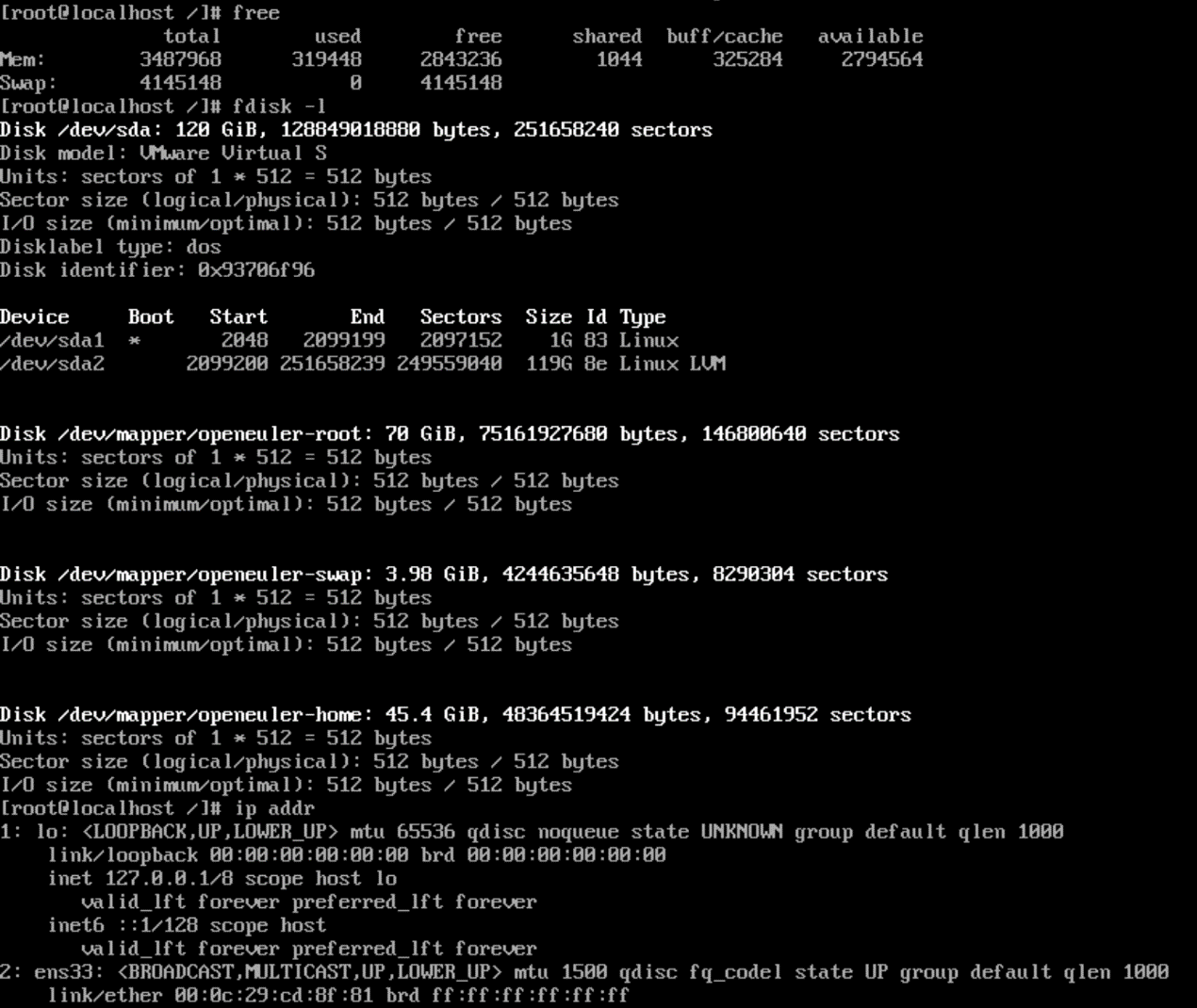
openEuler使用的命令风格和Linux系统是一样的，具体内容可以参考Linux命令。



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

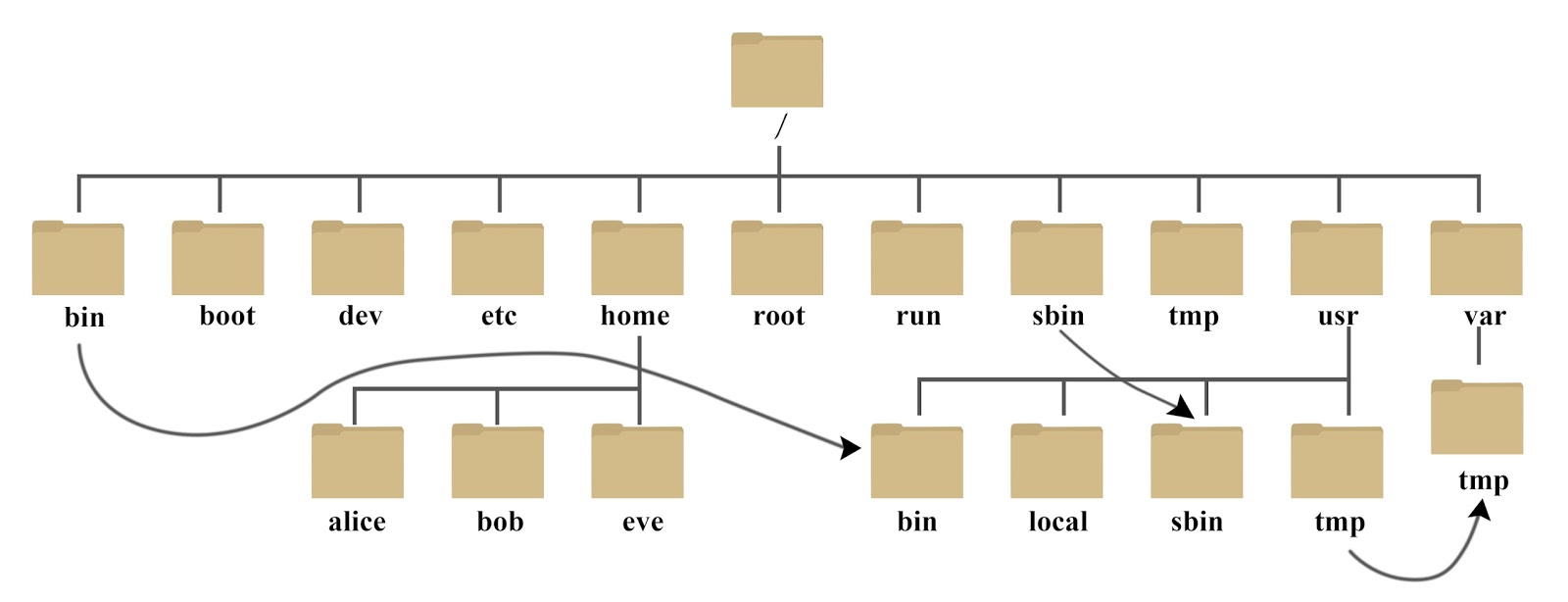


使用“free”命令查看内存信息，使用“fdisk -l”命令查看磁盘信息，使用“ip addr”命令查看IP地址：



**3.Linux系统目录结构与内核文件**

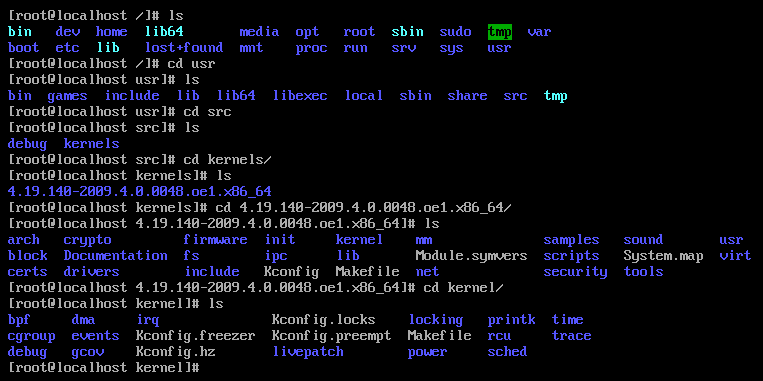
本专题实验以Linux内核为核心，主要实验内容包括设计新的系统调用、动态模块、同步机制、设计改进进程调度算法、设计改进存贮管理方式、设计自己的设备驱动程序和添加新的文件系统等，目的在于通过实验掌握操作系统低层的实现机理，能根据实际运行环境构造自己的操作系统内核，因此需要经常和内核打交道，而要查阅内核的代码并对其作出修改，我们首先需要对Linux系统的文件目录结构有一定的了解：



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 4.19.140-2009.4.0.0048.oe1.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 |

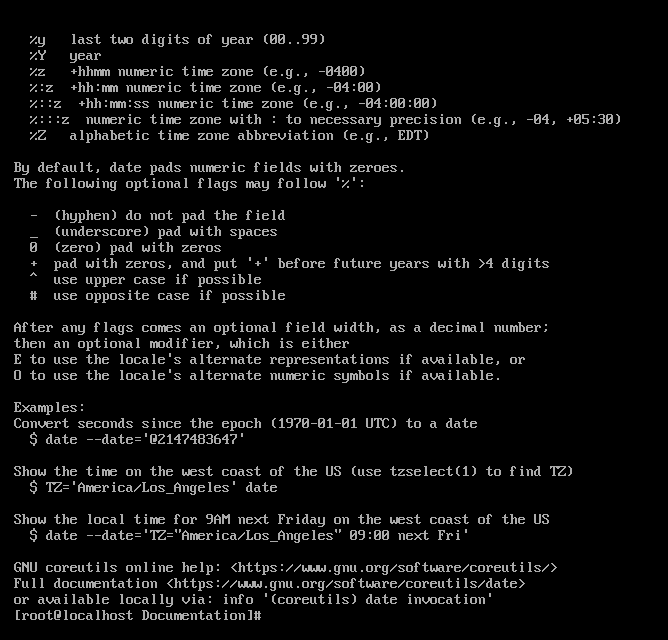
**4.3.用管道命令less在终端中翻页**

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：



输出结果如下：



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）：



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：



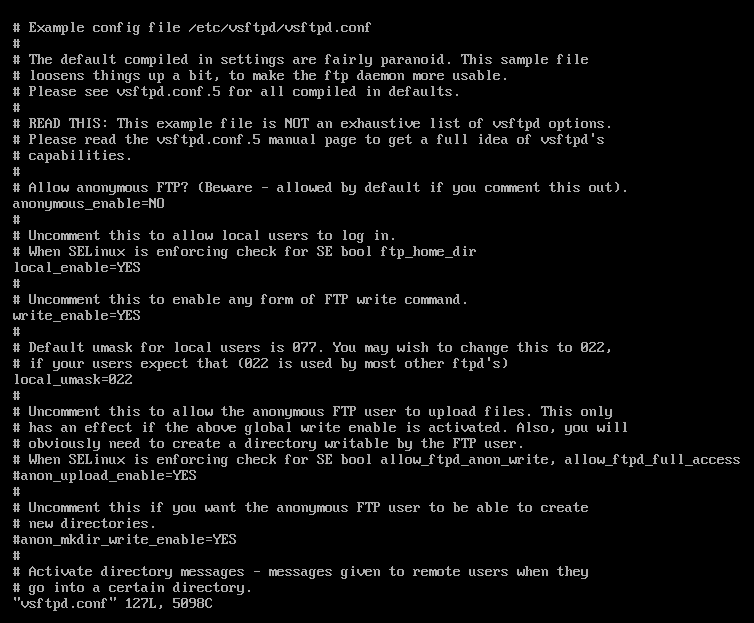
使用vsftpd需要先安装vsftpd软件，在已经配置yum源的情况下，通过root权限执行“dnf install vsftpd”命令，即可完成vsftpd的安装。接着执行“dnf install net-tools”命令，安装net-tools包使得netstat命令可可用，通过该命令查看通信端口21是否开启，如下显示说明vsftpd已经启动：

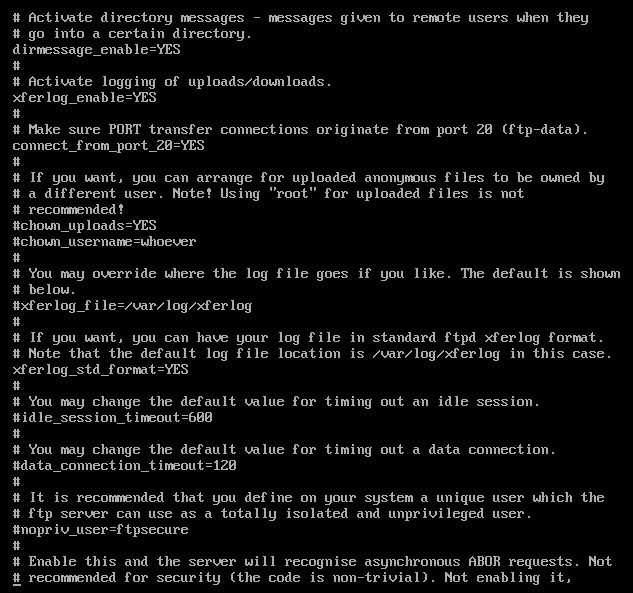


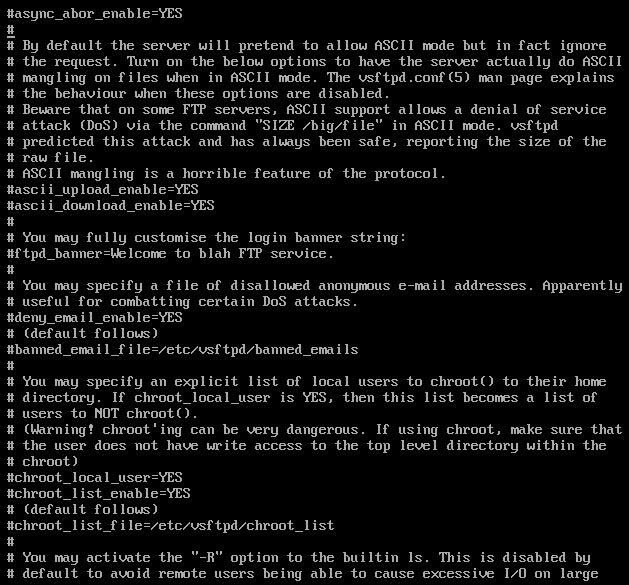
需要进行其它管理操作时，命令“systemctl stop vsftpd”和“systemctl restart vsftpd”分别可以停止和重启vsftpd服务，更细致的权限管理可以通过修改vsftpd的配置文件实现，配置文件的路径可以参考下表：

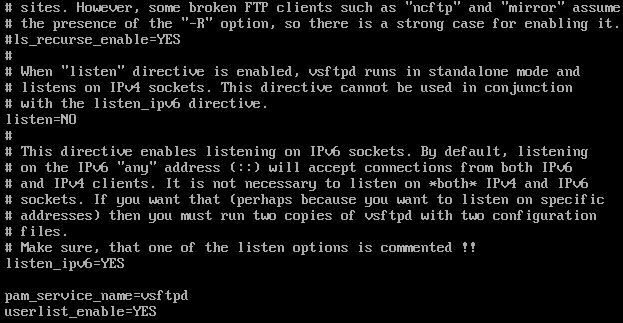
|  |  |
| --- | --- |
| **配置文件** | **含义** |
| /etc/vsftpd/vsftpd.conf | vsftpd进程的主配置文件，配置内容格式为“参数=参数值”，且参数和参数值不能为空。  vsftpd.conf 的详细介绍可以使用如下命令查看：  man 5 vsftpd.conf |
| /etc/pam.d/vsftpd | PAM（Pluggable Authentication Modules）认证文件，主要用于身份认证和限制一些用户的操作。 |
| /etc/vsftpd/ftpusers | 禁用使用vsftpd的用户列表文件。默认情况下，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：





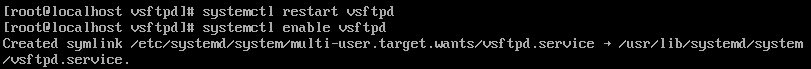




包含的参数和含义如下表所示：

|  |  |
| --- | --- |
| **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：



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

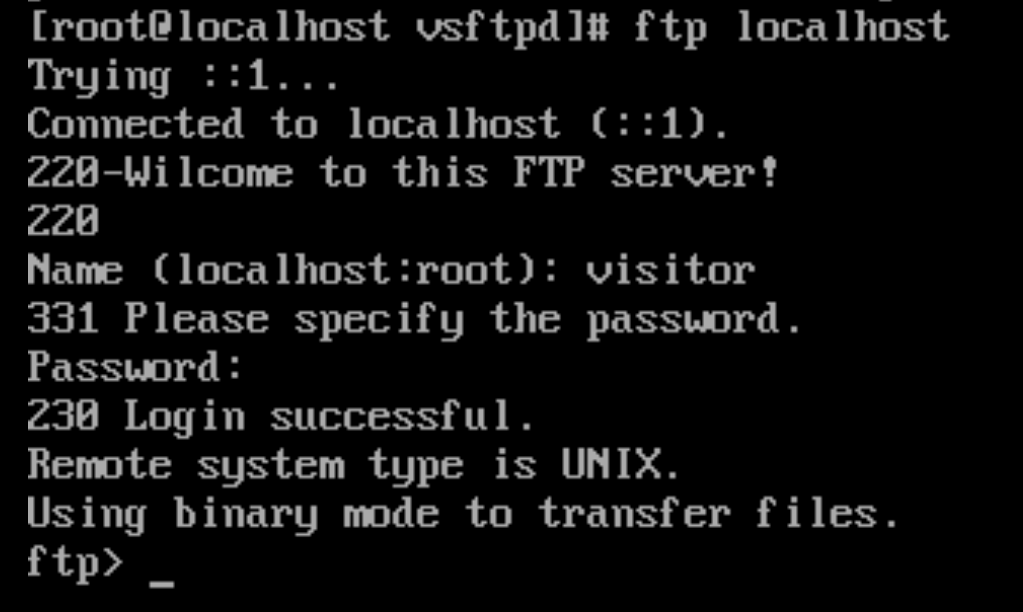
anon\_umask=022

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：



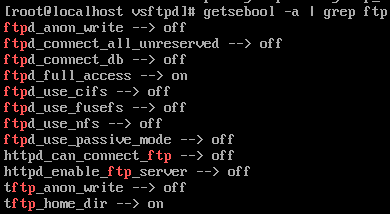
如果要将FTP开放给Internet使用，需要在root权限下对防火墙和SElinux进行设置：

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。