mjj版的linux入门教程

本文的首要目的是给予Linux初学者一个简单、易学的教程，以便在看完本文后对Linux系统有一个基础的认识（而非系统级的深入），可以对常见的软件和功能进行配置，甚至可以自己写一写一键脚本。

本教程写于2021年下半年，采用的系统为Debian GNU/Linux 11 (bullseye)。

0 前言吐槽CentOS

解释使用Debian而不是CentOS的原因

国内首批接触Linux系统的人主要集中在科研院校，大多数是延续了Unix-like的背景，在千禧年前后才有了真正意义上的Linux使用者：纯Linux平台开发、运行服务和应用，他们或直接或间接地推广了Linux系统。红帽（Red Hat, Inc.）在1994年就开始发布了同名的操作系统：Red Hat Linux（后改组为Red Hat Enterprise Linux，缩写为RHEL）。得益于红帽优秀的团队和商业支持，RHEL这一发行版迅速占领了国内市场。彼时的国内计算机市场远不如今日繁荣，在口口相传和红帽的推广中，RHEL成为了Linux入门的主流选项，即使后来号称用户友好的Ubuntu出现了，绝大多数尝鲜的人依然能看到众多网站里面只提供RHEL版本的教程。

CentOS是根据RHEL的源码重新编译的，等于换商标版本的RHEL，软件层面上，两者无本质区别。但CentOS是反人类的，至少是反入门用户的。使用RHEL的基本为商业用户，可以付费获得红帽的技术支持，或者干脆有一个自己的维护团队；而CentOS作为一个社区自发形成的操作系统，拥有落后的软件源/包，繁琐的配置，和对个人用户而言根本没有必要的SElinux等。举个例子，很多入门者修改SSH端口的时候，发现所有的操作都没有问题，但是死活无法生效，最终发现是没有在SElinux里面放行。如果你想安装个软件，你就得考虑是从落后主流版本好几代的软件源/包里面安装，还是自己下载源码进行编译以获取主流的使用体验。对于入门者而言，CentOS的安全性和稳定性是个虚假的概念，毕竟让一个刚接触Linux的人去自己编译源码安装，无异于让小学生上战场，输了就说是小学生战斗力太弱。

所以本文以Debian GNU/Linux（后续简称为Debian）来演示，也有着推广Debian的意思在里面，毕竟相比于Ubuntu往系统里面塞包括snap在内的一系列私货而言，Debain始终遵循着一个纯净的Linux的要求。而其他一些发行版，要么是专用性太强（如SUSE），要么是入门者不友好（如 Arch Linux），权衡之后，选择了写本文时，最新的Debian系统，即Debian GNU/Linux 11 (bullseye)。

1 环境搭建

1.1 系统选择与安装

Debian的安装包有一系列的前缀或者后缀，例如在默认的下载地址`https://www.debian.org/download`中的是`debian-11.0.0-amd64-netinst.iso`。其中，

- 11代表大版本是11，代号是bullseye，各版本代号都来源于电影《玩具总动员》中的角色名称；
- amd64是指系统为64位的，i386或者x86是32位的，amd64或者x86-64是64位的，32位系统已经被逐步弃用，目前仅在特定行业中使用；
- netinst是网络安装版本，只是个安装器，安装过程需要联网，而DVD后缀的是完整版（如：debian-11.0.0-amd64-DVD-1.iso），如果系统太大，会在DVD后面加数字，默认DVD-1是完整版本，其后数字的是软件源/包；

- 带firmware前缀的是包含第三方非开源驱动的（如：firmware-11.1.0-amd64-DVD-1.iso），其中就包含intel和Realtek等公司的闭源网卡驱动。

VPS全称为virtual private server（虚拟专用服务器），如果需要安装纯净版的Debian 11系统，推荐使用vicer的Linux一键重装脚本（如下）：

```
bash <(wget --no-check-certificate -qO-
'https://raw.githubusercontent.com/MoeClub/Note/master/InstallNET.sh') -d 11 -v
64 -p "自定义root密码" -port "自定义ssh端口"
```

1.2 常用的命令

`cat` 用于查看文本文件的内容，如 `cat /etc/os-release` 将显示系统信息，如下：

```
PRETTY_NAME="Debian GNU/Linux 11 (bullseye)"
NAME="Debian GNU/Linux"
VERSION_ID="11"
VERSION="11 (bullseye)"
VERSION_CODENAME=bullseye
ID=debian
HOME_URL="https://www.debian.org/"
SUPPORT_URL="https://www.debian.org/support"
BUG_REPORT_URL="https://bugs.debian.org/"
```

`touch` 新建文本文件，如 `touch /home/hello.py` 将在 `home` 文件夹下新建一个Python文件。

`ls` 列出所有文件，但默认只是显示出最基础的文件和文件夹，如果需要更详细的信息，则使用 `ls -la`，这将列出包括隐藏文件在内的所有文件和文件夹，并且给出对应的权限、大小和日期等信息。

`cd` 进入指定文件夹，如 `cd /home` 将进入 `home` 目录。返回上层目录的命令是 `cd ..`，返回刚才操作的目录的命令是 `cd -`。

`mkdir` 新建文件夹，如 `mkdir /home/Python` 将在 `home` 文件夹下新建一个 `Python` 文件夹。

`mv` 移动文件和文件夹，也可以用来修改名称，如 `mv /home/hello.py /home/helloworld.py` 将上文的 `hello.py` 重命名为 `helloworld.py`，`mv /home/helloworld.py /home/Python/helloworld.py` 将 `helloworld.py` 由 `home` 文件夹移动到了次级的 `Python` 文件夹。

`cp` 复制文件，`cp /home/Python/hellowrold.py /home/Python/HelloWorld.py` 将 `helloworld.py` 复制为 `HelloWolrd.py`。注意：Linux系统严格区分大小写，`helloworld.py` 和 `HelloWolrd.py` 是两个文件。如果想复制整个文件夹，则需要带 `r`，即 `cp -r`，但此命令无法复制隐藏文件夹，需要使用 `cp -r pathA/. pathB` 注意这个点 `.` 是灵魂。

`rm` 删除，即江湖传说中 `rm -rf`，`r` 为递归，可以删除文件夹中的文件，`f` 为强制删除。`rm /home/Python/helloworld.py` 可以删除刚才的 `helloworld.py` 文件，而想删除包括 `Python` 在内的所有文件，则是 `rm -rf /home/Python`。

`du -lh` 查看当前文件夹下，各文件、文件夹的大小，`l` 是硬链接（软连接类似于快捷方式），`h` 是让文件自动使用K/M/G显示而不是只有K。

1.3 基础文本编辑器nano、vim

Linux系统的一大优势（同时也是劣势）是默认不需要GUI，因此节省了大量的性能开支，无GUI版本的Debian 11可以在512M甚至更小内存的VPS上正常启动和运行。但缺少GUI加大了入门者修改文件的难度，所幸Debian 11自带了简便易用的nano文本编辑器。以下以修改系统的更新源为例

```
nano /etc/apt/sources.list #打开sources.list文件，在Linux系统中，#是注释符，其后的内容会被忽略
```

如图所示，即为`nano`打开`sources.list`后的界面，最下面两行为提示，比如`Ctrl+E`为退出，如果文档被改动了，则会出现下图，询问是否保存。如果没有被更改，则会直接退出。

`Y`则保存，`N`则不保存，`Ctrl+C`取消操作。此处输入`Y`，则会如下图：

此时按下`Enter`键就会保存了。

这里多提一句关于Debian 11的更新源内容，一般是以下6行。

```
deb http://deb.debian.org/debian bullseye main contrib non-free
deb-src http://deb.debian.org/debian bullseye main contrib non-free

deb http://deb.debian.org/debian-security/ bullseye-security main contrib non-free
deb-src http://deb.debian.org/debian-security/ bullseye-security main contrib non-free

deb http://deb.debian.org/debian bullseye-updates main contrib non-free
deb-src http://deb.debian.org/debian bullseye-updates main contrib non-free
```

deb表示为已经编译好的安装包，类似于Windows上的MSI安装包，deb-src是源文件，万一没有打包好，提供自己本地编译安装的机会。总共分三大行，第一行是系统主文件，第二行是安全性更新，第三个是一些更新补充，推荐三个都写上。在每行的末尾都有`main contrib non-free`字样，其中`main`是官方给的包/源，严格遵守相关开源协议；`contrib`是包/源本身遵守相关开源协议，但是它们的依赖则不是；`non-free`是私有的软件，比如上文提到的Realtek的WiFi驱动等。除此之外，其实还有个`Backports`作为第四大行，是将比较陈旧的软件移植过来的，很少会用到，一般默认不写上。

nano虽然好，但是功能简单，只适合一些简单的文本文件编辑功能，而发展自vi的vim则被成为编辑器之神（Emacs被称为神之编辑器，Linux之父Linus Torvalds就在用）。系统会自带vi但是不带vim，正好我们可以使用上述修改过的更新源来安装vim作为示例。

```
apt update # 更新一下源
apt install vim -y #安装vim这个软件 -y是确认安装
```

使用`vim /etc/apt/sources.list`打开更新源文件，如下图所示：

vim功能众多，使用复杂，得慢慢说。左下角是此文件的路径和名称，右下角是光标此时的行数和列数。此时是无法直接输入，要先按下`insert`或者`i`键变成插入模式才行。此时，左下角如下图，变成了INSERT/插入模式。

然后就是该怎么写就怎么写，一些快捷键去百度谷歌必应吧，说的肯定比我详细。但是必须提到如何保存文件：`insert`模式下按`esc`键（一般是键盘最左上角，99%的人可能都不怎么用的一个键），INSERT会消失不见，如下图：

这个时候再按下`:`键，界面上也会出现一个冒号，如下图。注意，这个冒号是半角的，全角冒号是没用的。

这个时候，按下`wq`这两个键，即可保存内容。w是write/写入的意思，q是quit/退出的意思。如果你不想保存，则只输入q键即可，但是有时候因为文件已经被修改了，vim不让退出，这时候输入`q!`就可以了，感叹号是强制执行的意思，执行后文件不会被修改并且会退出vim。

1.4 更新系统

至此，不管是使用nano还是vim都可以对更新源进行编辑了，让我们来具体了解一下如果更新系统和相关指令。

```
apt update
apt list --upgradable
apt upgrade -y
```

以上三行，分别是和更新源同步，显示出哪些软件可以更新，以及进行更新。

如上文中，安装了vim，若想卸载vim，则有以下两个命令，任意一个即可，但之间存在差别。

```
apt remove vim -y
apt purge vim -y
```

第一个会址卸载vim软件本身，配置文件仍然会本留下；第二种连带着配置文件和相关依赖一起卸载了，所以存在一定风险。除此之外，`apt autoremove`是对整个系统进行整理，将不需要的依赖卸载了，不针对于特定软件。

2 SSH连接和基础配置

一般VPS供应商都会提供SSH的链接方式，包括用户名，密码和端口号，一些注重安全性的会修改端口号甚至只有采用密钥才能登陆VPS。这里使用纯净版的系统和默认配置进行演示。

2.1 连接SSH的软件和相关操作

SSH软件有开源的和不开源的，有付费的和免费的，整理了一个常见SSH客户端（Windows平台）的对比表格和相关信息。其实在2021年，macos、Linux和windows 10都自带SSH功能，这里先不讨论。个人目前主用mobaxterm，偶尔使用xshell。

名称	免费与否	下载地址
Xshell	家庭/学校免费	https://www.netsarang.com/zh/free-for-home-school/
MobaXterm	家庭版免费	https://mobaxterm.mobatek.net/download.html
FinalShell	基础功能免费	https://www.hostbuf.com/t/988.html
electerm	免费+开源	https://github.com/electerm/electerm/releases
PuTTY	免费+开源	https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html

Xshell：传播广泛，自带中文，个人使用完全免费，但是会话窗口限制最多只能打开四个SSH连接，再多之后就会自动新建会话窗口了，传输文件需要配合Xftp才行。目前国区被臭名昭著的思杰马克丁代理了，如需使用，请前往官网下载，在输入邮件和姓名后，会收到一封邮件，邮件里面给出下载连接。

MobaXterm：只有英文版本，偶尔会反应慢半拍，除此之外没缺点。功能极其强大，传输文件、性能监控、串口通信、X11支持、IP检测、宏、WSL、远程桌面等，能想到的功能都有，而且个人使用免费。

FinalShell：国人开发，所以本地化很好，全中文，日常需要的功能也都有。有一些进阶功能需要付费，也可以云端保存SSH账号，虽然也是付费功能。缺点的话，和MobaXterm一样，JAVA写的东西，总是让人觉得慢半拍。

electerm：日常所需功能都有，完全开源和免费，还可以通过GitHub实现免费的云端保存SSH账号功能，适合自己折腾和魔改。基于electron开发的，从而实现了跨平台，Windows、Mac和Linux都有客户端。不过缺点也显而易见，electron本质上是个浏览器，占内存和硬盘空间。

PuTTY：由Simon Tatham开发和维护的，老牌中的老牌，但是缺少人性化设置，不推荐。

2.2 SSH配置文件介绍和修改

SSH的配置文件在`/etc/ssh/sshd_config`中，是一个纯文本文件，可以使用`nano`或者`vim`打开和编辑。打开文件后，在前几行就能看到`#Port 22`字样，这个代表使用了默认的22端口作为SSH连接使用。因为大家都在使用22端口，所以会有一些扫描机器使用弱密码不断尝试登录，使用`lastb`命令可以查看登录失败的记录，如下图。233333是尝试登录的账号，144.214.xxx.xxx是发起者的IP，最后面是尝试登录的时间。

因此，我们可以改成高端口，比如`35261`这种没有特殊含义/排列的随机数，以减小被攻击的可能。要注意端口只能在0-65535之间，并且很多低位数的端口，已经被共识的程序占领了，比如80端口是http的，443端口是https的，就如22是SSH的一样。此处，我们修改`/etc/ssh/sshd_config`中的端口数后，还需要重启SSH服务才行，否则只会在系统下次重启后才启用新的端口。

```
systemctl restart ssh #重启SSH服务
```

`systemctl`是systemd的命令，用于启动和监控系统服务的，在系统内核启动后，systemd就会开始服务，`restart`即重启的意思。关于systemd的相关内容，后文会详细说明。

除此之外，把密码改的复杂一些，也可以有效的降低系统被黑的风险，使用`passwd root`命令，即可修改root账号的密码，会提示`New password:`，此时输入新密码，注意这里是看不见任何输入反馈的，随后在显示`Retype new password:`后再次输入一遍，如果两次密码相同，就会更新root密码了。

2.3 使用密钥登陆SSH

即使更改了端口，但因为使用密码即可登录，考虑到不是所有人都会使用强密码，所以SSH提供了使用密钥登录的功能，可以简单理解成是一长串复杂的并且可以相互验证的密码。以root用户为例，演示如何将SSH由密码登录改成密钥登录。

输入`ssh-keygen -t rsa`，随后一路enter键，如下图

在显示完成后，在`/root/.ssh/`文件夹下，你将看到`id_rsa`和`id_rsa.pub`两个文件，id_rsa是私钥，下载下来并妥善保存，id_rsa.pub是公钥，放在服务器上的。将id_rsa.pub写入到SSH的密钥文件中：

```
touch authorized_keys
cat id_rsa.pub >> authorized_keys
```

除此之外，还需要给文件和相关文件夹合适的权限：

```
chmod 600 authorized_keys
chmod 700 ~/.ssh
```

这里有个`~`，它代表的是`当前用户`，比如现在是root用户，那`~`就是`root`，所以`chmod 700 ~/.ssh`等于`chmod 700 /root/.ssh`

现在密钥已经配对好了，还需要修改SSH的配置文件，打开`/etc/ssh/sshd_config`文件，查找并修改如下：

```
PubkeyAuthentication yes # yes表示允许密钥登陆
AuthorizedKeysFile      .ssh/authorized_keys .ssh/authorized_keys2 # 指定密钥的文件位置，这里是去掉了开头的#
PasswordAuthentication no # 不允许使用密码登陆，等测试密钥登陆成功了再修改此条，以防无法登陆
```

使用`systemctl restart ssh`重启SSH服务，此刻，你将只能使用密钥才能登录，一旦私钥遗失了，就再也进不去了。

3 Linux文件系统

3.1 文件系统格式

与Windows分割硬盘（甚至一个硬盘被划分成了好几个）不同，Linux的是将所有硬盘都`挂载`在了一起。简单来说，Windows分C盘D盘等，还针对软盘额外给予了A盘和B盘，Linux把所有的硬盘都放在了`/`下，即`根目录`，这也是Linux中root账户的权限最大的原因，`root`即为根，如同树根一样，所有的内容都要基于根才有了可能。文件系统是另外一个极其复杂的内容，这里只提到Windows使用的是NTFS，而Linux普遍采用EXT4格式，这两种文件系统互不兼容，装在Linux系统上的硬盘，在Windows上是无法直接读写的，必须使用额外的软件才能访问。反过来，在Linux上读取Windows下的硬盘中的内容，需要安装`ntfs-3g`才行。这两种文件系统各有优缺点，一般人用就行了，不要问，问就是用默认。实际上，目前在广泛使用的文件系统种类繁多，所有需要在它们之间交换文件的时候，会使用exFAT格式的U盘/硬盘（exFAT是FAT的替代品，因为FAT下单个文件最大不能超过4G）。

3.2 文件树、文件夹功能和权限

在`/`目录下，使用`ls -la`会显示出所有的文件和文件夹（如下图所示），`a`是列出所有文件，`l`是显示详细信息。

第一列是文件/文件夹的权限，一共有10个字符，第一位是文件类型，比如d代表文件夹，l代表链接。之后，三个为一组，总共3组。r是读，w是写，x是执行，也可以通过数字来区分，r是4，w是2，x是1，所以有了常见的一把梭`chmod +777`。第二列是硬链接数量，即这个文件/文件夹下有多少真实放着的文件。第三列和第四列是这个件分别属于谁，以及这个人是哪个组的。这里的`组`概念来源于最开始的unix是个多用户系统，所有会把用户分类，比如某软件用户放一个组，系统维护人员放一个组等。第六列是文件/文件夹大小，默认单位是K。第七八九列是修改的日期。最后一列是文件/文件夹的名字。我们会发现有一些`->`的字样，这是指软链接。软连接类似于Windows上的快捷方式，而硬连接类似于复制了一份（但并不会真的占用空间）。

- bin或者usr/bin：应用程序，比如Python的主程序就在这里
- boot：系统启动文件
- dev：外部硬件设备，Linux下一切皆文件，所以外部硬件设备也是以文件形式出现
- etc：系统的配置文件，比如上述提到的SSH的配置文件就在这里
- home：用户目录，类似于Windows上的桌面
- initrd.img：启动文件，可以看到它被软连接到了boot目录中
- lib：库文件，类似于Windows的dll，程序的依赖都在这里
- lost+found：丢失寻找文件，系统被强迫关机后，会在这里记录下来
- media：媒体文件，如果系统发现了光盘之类的，会自动挂载到这里
- mnt：临时挂载目录，上述的光盘，还有U盘硬盘，如果手动挂载，都会选择这里
- opt：系统额外软件的安装位置，极少使用，比如甲骨文的数据库会放一些东西在这里
- proc：系统进程/内核会把一些信息放到这里，本质上是反应系统状态而不是文件
- root：root用户的“桌面”，普通用户在home中
- run：系统启动后存放临时文件
- sbin：root用户的"bin"
- srv：放服务运行而需要的文件
- sys：文件系统，里面包括进程信息，设备信息和终端信息

- tmp：临时文件
- usr：共享资源，类似于Windows安装软件的默认目录
- var：不断变化的文件会放在这里，比如日志
- vmlinuz：启动文件，可以看到它被软连接到了boot目录中

如果我们自己写了一个程序，还放在系统里面运行，那一般是在`/usr/local/`中新建目录，这遵循着Linux系统的默认规则。

3.3 示例：挂载U盘

如果是Ubuntu桌面的话，会自动挂载U盘，但是无GUI版本的Linux大概率不会，所以会需要手动挂载，又或者加了一块新的硬盘，需要我们自己挂载。

```
fdisk -l
mkdir /mnt/usb
mount /dev/sda1 /mnt/usb
umount /mnt/usb
```

`fdisk -l`是显示出所有的储存，会显示出来类似于`/dev/sda1`等，`mkdir /mnt/usb`在`mnt`目录里面新建一个文件夹，即挂载点，假设`sda1`就是我们插入的U盘，`mount /dev/sda1 /mnt/usb`将这块U盘挂载到了`/mnt/usb`中，这时候我们就能在`/mnt/usb`中看到U盘里的文件。如果不再需要了，要手动移除这个U盘，使用`umount /mnt/usb`命令。

4 Shell/Dash入门

让人头大，Shell本身就能写一本书了，少说得有300页！这里面夹杂着从Unix开始的一大堆事情，shell的发展，bash和dash的区别与联系，本身的命令，调用系统的命令，交互方式。累了，姑且先把它当成一堆命令拼凑起来的脚本吧。

还是写个例子，简单介绍一下实际内容，等以后有时间了再继续补充。比如我们想写一个查看CPU和内存使用率的脚本：

```
#!/bin/bash

echo "which useage do you want to konw?"
echo "1 for CPU, 2 for RAM"
read choice

if [ $choice -eq 1 ]
then
   echo "CPU usage"
   grep 'cpu ' /proc/stat | awk '{usage=($2+$4)*100/($2+$4+$5)} END {print usage
"%"}'
elif [ $choice -eq 2 ]
then
   echo "RAM usage"
   free -m | grep Mem | awk '{print ($3/$2)*100 "%"}'
else
   echo "WRONG INPUT"
```

```
fi
```

细说每一行内容

`#!/bin/bash`是指定此文件由/bin下面的bash这个程序来执行。

在Debian 11里面，bash其实是dash，别问dash是什么，就写bash，天王老子来了也写bash。Bash全称是GNU Bourne-Again Shell，bash被从NetBSD（一个Unix的分支）上移植到Debian上，所以叫dash (Debian Almquist Shell)。

`echo "which useage do you want to konw?"`是输出冒号内的文字

```
echo "1 for CPU, 2 for RAM"
read choice
```

把输入内容赋值给`choice`这个变量，即数字1或者2。`rcho -p "1 for CPU, 2 for RAM" choice`也可以实现相同功能。

```
if [ ... ]
then
    ...
    ...
elif [ ... ]
then
    ...
    ...
else
    ...
fi
```

这是一个if...elif...else的判断语句，先经过两次判断，如果都不能成功，那就执行最后一行。

`$choice -eq 1`把刚才的`choice`这个输入变量和数字1对比，注意，要有`$`才代表变量，不然就默认是文字，`-eq`是等于的意思。判断是否等于1，是的话就给出CPU使用量，如果不等于1，那就继续判断是否等于2，是的话就给出RAM使用量，如果不等于2，那就输出错误提醒，然后结束。`grep`是抓取有关键词的那一行，`$2`是这一行的第几个内容，如下：

`grep Mem`抓取到了第二行，即真实内存这一行，`$2`是内存总量，`$3`是已经使用了的内存，因此`($3/$2)*100`就是已经使用了百分之多少的内存，

5 Crontab定时任务

Crontab用于定时任务，比如设定周五晚上运行脚本备份网站，又或者每分钟检查一下CPU使用率等。但除此之外，crontab还有个`@reboot`功能，即可以在系统启动的时候自动运行指定程序。

推荐`crontab -e`，其中的`-e`是指当前用户，不建议直接使用`crontab`。首次运行`crontab -e`的时候，会让选择使用何种编辑工具，这个随便，nano和vim basic都行，什么顺手和习惯就用什么。

如上图，将每隔15分钟，就会使用位于`usr/bin`中的`python3`运行位于`/usr/local/weather`中的`weather.py`程序。前五个星号其实是设置的时间，推荐去 https://crontab.guru/ 这里直接设置时间（如下图）。第一个星号是分钟，第二个星号是小时，第三个星号是天，第四个星号是月份，第五个星号是每周的第几天。

6 系统权限

6.1 root和user，以及sudo

上述已经简单的减少了root来源，由于root的权限太高，以至于在实际使用中发现并不安全，而且作为一个初始目的是多用户多终端的操作系统，Linux主要操作都不需要发生在root用户上的。所以这里就有了user这个角色，如果用户多了起来，为了便于管理，也会把某些用户分组，就有了group的概念。以下演示使用root用户新建一个user用户并进入此用户：

如图所示，`adduser mjj`为新建一个叫做mjj的用户，由于此前并没有除了root之外的用户，所以会使用这个名字作为group/组的名字，并且在/home文件夹里面生成一个mjj文件夹，即mjj的"桌面"。所以输入两次密码，之后会问一堆问题，都是例行的，一路enter就好，最后会问一下信息对不对，输入y就完成添加新用户了。

但此时，mjj这个用户的权限是很小的，四舍五入等于没有，连某些文件夹都不能进去更别说执行软件了。使用`su mjj`切换到mjj用户中，可以在终端中看到已经从root@rn变成了mjj@rn，rn是这台服务器的名字，即为某某在rn这台服务器上。查看以下root文件夹下有些什么东西，结果发现权限不够而被拒绝访问/Permission denied。

所以我们要给一个能够临时使用root权限的能力，这被称为`sudo`。

```
su root #切换回root账号
apt install sudo -y #有些时候，纯净安装的Debian系统是没有sudo的，所以要安装一下
usermod -aG sudo mjj ##给予mjj用户sudo权限
```

此时，我们再切换到mjj用户上，在刚才的命令前加上sudo，临时获取root权限，就可以查看了：

在用户首次使用root权限的时候，系统会提示三个准则，也请谨记：

1. 尊重他人隐私；
2. 输入之前请三思；
3. 能力越大，责任越大。

6.2 chmod和chown

chmod的全称是change mode，是针对于文件夹或者文件，改变它们的权限，这样就可以让某些用户正常使用了。这里不深入探讨chmod的使用详解，仅演示一些常见的内容。`chmod +x helloworld.py`这里的`x`在上面说过，是执行的意思，即赋予此程序被执行的权限，多见于一键脚本里面，让脚本能够正常运行。`+`是新增权限，如果是`-`则是去除对应的权限。`chmod -R 755 folder/` `-R`如上述的`rm -`

`rf`中的`r`一样，是递归，即从这里开始，下面不论多少层文件夹，都执行这个命令。`755`是换算下来，则是root用户可以读写执行（1+2+4=7），用户和用户组只能读和执行（1+4=5），不能对文件进行更改。某些程序会对文件的权限有着极其严重的控制，比如上说的SSH密钥，分别给`authorized_keys`600的权限和`.ssh`文件夹700的权限，意味着只能被所有者/owner读写，在例子中即为只能被root账户读写。这种设计让没有相应权限的人无法修改密钥登陆的方式，换而言之，隔绝了用户之间的操作，从而增强安全性。

chown的全称是change owner，是用于设置文件所有权的。由于`归属者`的概念并没有文件这个概念常见，所有chown比较少见到，大多数人接触到的时候，大概是建网站的时候用`chown -R www-data:www-data-group /var/www/html`来确定文件关联。这句的意思是，将`/var/www/html`这个文件夹及里面的所有文件都归给`www-data-group`用户组的`www-data`用户。这样做的目的是实现权限分离，文件分离，从而让服务器可以更方便的被维护，以及明确使用途径。不过考虑到mjj大部分都是使用VPS的，可能很难遇到需要chown的情况吧。

不要`chmod +777`！不要`chmod +777`！不要`chmod +777`！**人才是服务器安全的最大漏洞！**

7 Systemd入门和配置

7.1 开机自启和进程守护

Systemd是由Redhat家的Lennart Poettering开发的，其人以创造性和不靠谱闻名，Systemd在最开始的时候，和init相比没有明显优势，经过多次迭代才有了今天的稳定性和适用性，现在就让老旧的init进入历史垃圾桶吧。事实上，在Linux系统启动的时候，一旦kernel运行了，Systemd就会跟随启动，之后由Systemd唤醒并维护各个程序的正常运行，比如网卡，显示器，SSH服务等。你会在`/etc/systemd/system/`文件夹中发现一个叫做`sshd.service`的文件，并且还是enable模式的，这意味着SSH是`开机自启`的，并且系统会一直监控这这个程序，如果程序崩溃，系统会尝试自动重启它以确保能够正常运行。

以著名的内网穿透`frp`的服务器端的Systemd文件为例（下节将详细介绍如何搭建frp），将`frps.service`放到`/etc/systemd/system/`文件夹中，使用以下命令

```
systemctl enable frps.service
systemctl start frps.service
systemctl status frps.service
systemctl restart frps.service
```

systemctl是systemd在系统中的程序名字，enable是指让这个程序能够开机自启，start为让程序现就运行，status是查看这个程序现在的状态，restart是重启程序。

当然，我们也可以自己写systemd的service文件，这里以 https://github.com/cnsilvan/UnblockNeteaseMusic 解锁网易云音乐的程序做参考

```
[Unit]
Description=UnblockNeteaseMusic
After=network.target
Wants=network.target

[Service]
Type=simple
WorkingDirectory=/usr/local/UnblockNeteaseMusic
ExecStart=/usr/bin/node app.js -e http://music.163.com -s -p 8888
```

```
RestartPreventExitStatus=23
Restart=always

[Install]
WantedBy=multi-user.target
```

一共分为三组，分别为Unit，Service和Install。Unit是这个服务的名称，示例中为`UnblockNeteaseMusic`而`After`和`Wants`指明的`network.target`意味希望这个程序在网络服务启动后再启动，毕竟是个网络功能，不能还没有网就启动了。Service是核心部分，`Type`指定了类型,`Simple`是默认的类型，发现有网了就启动。此外，常见的还有`fork`和`idle`，前者意味着程序依赖于另外一个程序的运行，通常还会配置`PIDFile`，后者是等系统空闲了再启动，属于一点都不急的。`WorkingDirectory`是工作目录，`ExecStart`是执行的命令，实例中，是用位于`/usr/bin/`的nodejs执行位于工作目录的`app.js`这个文件，并且还带了参数`http://music.163.com -s -p 8888`。`RestartPreventExitStatus`是指如果报错信息为`23`则不会再重启了，具体报错信息是运行的程序决定的。`Restart=always`指只要不是23的报错信息，那就一旦服务停了，Systemd就会去重启。最后一部分，Install中的`WantedBy=multi-user.target`指网络服务已经正常启动，也可以让用户登录了，但是并没有开启GUI服务，这个部分不用去探究。

7.2 Timer代替Crontab

我写了一个自动登录百度贴吧并签到的shell脚本，想每天都运行一次帮我拿积分，但是又不想用crontab实现定时任务，那么Systemd也是有类似的功能的，名字叫做Timer，即定时器。这个功能需要两个文件，比较繁琐。

需要再`/etc/systemd/system/`中写两份配置文件，`tieba.service`和`tieba.timer`，前缀必须一样，后缀不同。

前者很简单，就是个脚本（如下），名字和程序的路径：

```
[Unit]
Description=Tieba Sign

[Service]
ExecStart=/home/tieba.sh
```

后者`tieba.timer`需要详细解释：

```
[Unit]
Description=Tieba Sign Timer

[Timer]
OnCalendar=*-*-* 12:00:00

[Install]
WantedBy=timers.target
```

Timer的名称需要是service名字后面加一个Timer，用以提高准确性。`OnCalendar`类似于`corntab`的`* * * * *`，实例中的意味每天中午12点的时候执行以下`tieba.service`中的位于`/home`文件夹的`tieba.sh`这个程序。此处的`WantedBy`是`timers.target`，指明是个定时器。

8 手动配置系统：以frp为例

frp是个内网穿透软件，可以将局域网设备通过frp服务端映射出来，实现公网服务，常见的有SSH、http/https服务等。这里以将内网设备的SSH映射到有公网IP的服务器上为例，从而不在家也能服务家里面的服务器了。

在这里下载最新版的安装包 https://github.com/fatedier/frp/releases

如图所示，0.38.0是版本号；后面的是系统，darwin是MacOS，freebsd是UNIX的一个分支，这里我们选择linux；紧跟着的是CPU架构，由于本次服务器的客户端是装了64位系统的树莓派4B（arm架构的CPU），所以选择`frp_0.38.0_linux_arm64.tar.gz`，而服务端是普通的VPS，所以选择`frp_0.38.0_linux_amd64.tar.gz`。

服务端配置

注意：版本号和CPU架构须按照实际情况决定。

```
wget
https://github.com/fatedier/frp/releases/download/v0.38.0/frp_0.38.0_linux_amd64
.tar.gz
# 使用wget下载软件包
tar -zxvf frp_0.38.0_linux_amd64.tar.gz
# 解压下载的软件包
cd frp_0.38.0_linux_amd64/
# 进入解压后的文件夹
mkdir /etc/frp/
# 新建一个frp的文件夹
mv frps.ini /etc/frp/frps.ini
# 把服务器端的配置文件放到刚才新建的文件夹
mv frps /bin/
# 把服务器端软件放到/bin中
mv systemd/frps.service /etc/systemd/system/frps.service
# 放置Systemd文件
systemctl enable frps.service
# 设置开机自启
systemctl start frps.service
# 立即运行
```

客户端配置

```
wget
https://github.com/fatedier/frp/releases/download/v0.38.0/frp_0.38.0_linux_arm.t
ar.gz
tar -zxvf frp_0.38.0_linux_arm.tar.gz
cd frp_0.38.0_linux_arm/
mkdir /etc/frp/
mv frpc.ini /etc/frp/frpc.ini
mv frpc /bin/
mv systemd/frpc.service /etc/systemd/system/frpc.service
```

上述与服务器配置类似，就不重复了，但需要额外修改服务端的配置文件，让它知道该和谁连接，打开`/etc/frp/frpc.ini`配置文件

```
[common]
server_addr = 服务器ip
server_port = 7000

[raspi]
type = tcp
local_ip = 127.0.0.1
local_port = 本地服务器的SSH端口
remote_port = 远程端口
```

其中，需要填写服务器端的IP，7000端口是握手和保活用的，默认就好了。`[raspi]`是客户端的名字，不可以重复，`local_port`是客户端的SSH端口，`remote_port`是远程的端口，此处假设是6000。

```
systemctl enable frpc.service
systemctl start frpc.service
```

设置开机自启并立即运行，此时在SSH软件上，通过`服务器IP:6000`就可以连接到这台内网的树莓派了。

9 网站环境搭建

网站搭建，说简单也简单，安装一个nginx放个html页面就算是了，但也可以做的极其复杂以至于需要一个团队，比如淘宝。这里提供了两个搭建网站的方法：面板和手动搭建。对于小白用户，还是推荐用用面板吧，不然出问题，网站被黑都不知道如何解决。

9.1 宝塔解人忧

宝塔面板是个伪开源的一键式建站面板，国内版可以在 https://www.bt.cn/ 中找到安装方式，目前的安装命令是`wget -O install.sh http://download.bt.cn/install/install-ubuntu_6.0.sh && bash install.sh`。需要注意的是，国内版需要登陆并且验证手机号后才能操作，宝塔也有强制后台升级的前科。

除此之外，还有国际版的叫做aapanel，安装地址为 https://www.aapanel.com/install.html ，安装命令是`wget -O install.sh http://www.aapanel.com/script/install_6.0_en.sh && bash install.sh`。相对而言，国际版的隐私保护会更好一些，不会要求手机号等信息，但是默认语言是英文，如果会哪怕一点点英文，都推荐使用国际版的。

宝塔有一点不好的地方是动辄编译（原先是在CentOS上开发的，所以有这个臭毛病），面板是python3写的，安装起来很快，但是要安装一些服务的话，如果VPS性能不好，则需要花费相当长一段时间来编译安装，普遍30分钟起步。

9.2 手动搭建

宝塔面板是将网站搭建可视化了，本质上和手动搭建没有区别。这里主要是介绍常见的相关软件和Let's Encrypt配置SSL证书的方法。

9.2.1 Apache和Nginx

Apache和Nginx都是Web服务器。前者是老牌Web服务器软件，对PHP有着优秀的支持，并且动态响应优秀，但是对性能和内存要求高。后者是俄罗斯出品，对静态网站支持良好，性能消耗也更小，反代和简单的网站都倾向于使用Nginx，甚至还有一系列基于Nginx衍生出来的版本，比如Tengine就是淘宝从Nginx衍生出来的，用以支撑淘宝的各种服务。除此之外，还有个Caddy也用的比较多，这是由golang语言写出来的，所以对多线程高并发的支持很好，并且自带SSL证书申请的功能。

入门用户首选Nginx，毕竟网站没有什么服务，对VPS的性能消耗也少点。当然，选择Apache也完全可以，入门用户其实很难用到需要对比选择Apache和Nginx的时候。Caddy的话，先不推荐入门用户了。

安装Apache使用以下命令

```
apt install apache2
```

安装Nginx使用以下命令

```
apt install nginx
```

这两个软件安装完后，都会开机自启和立刻运行，浏览器中输入`http://ip`就可以看到默认的网页，比如Nginx的是这样：

9.2.2 PHP

安装了Nginx之后，可以实现静态网页，但是常见的网站平台，比如Wordpress和typecho都是PHP写的，所以还需要安装PHP才能运行。

```
apt install php-fpm
```

安装完php还不算完事，还得让Nginx知道，也就是需要更改Nginx的配置文件。

进入目录`/etc/nginx/sites-available/`中，将默认的文件`default`重命名为网站的域名，比如`mjj.hostloc.com`，即`mv default mjj.hostloc.com`，打开`mjj.hostloc.com`，在下述的第二行末尾加入`index.php`

```
# Add index.php to the list if you are using PHP
index index.html index.htm index.nginx-debian.html index.php; # 注意加入了
index.php
```

并且将下述的`_`改成网站域名

```
server_name _;
# 改成 server_name mjj.hostloc.com;
```

随后，重新软链接，并重启Nginx

```
rm /etc/nginx/sites-enabled/default
ln -s /etc/nginx/sites-available/mjj.hostloc.com /etc/nginx/sites-
enabled/mjj.hostloc.com
systemctl restart nginx
```

注意域名不要填错了，重启完后，Nginx将能够和PHP一起支持动态网站。

9.2.3 MySQL和MariaDB

MySQL是一个市场占有率极大的数据库软件，应用场景极其广泛，最开始是SUN公司开发的，2009年被甲骨文收购。甲骨文作恶多端，所以MySQL的一部分作者则独立出来，直接做了一个复刻版的，被命名为MariaDB，Maria是作者女儿的名字，Linux社区逐步放弃MySQL而采用MariaDB。所以在近期发布的各Linux版本中，默认是没有MySQL的，一律采用MariaDB。

```
apt install mariadb-server # 安装数据库
mysql_secure_installation # 首次配置
```

由于是首次使用，所以在如下提示中，直接enter键就可以了，因为数据库的root用户此时并没有密码

```
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user.  If you've just installed MariaDB, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.

Enter current password for root (enter for none):
```

随后，会询问是否要设置数据库的root密码，怎么说呢，反正就建个站而已（不涉及多用户多服务），有没有无所谓，我习惯性的不设置（输入N）

```
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MariaDB
root user without the proper authorisation.

Set root password? [Y/n] N
```

现在，新建一个用户和对应的数据库

```
mysql # 进入数据库，如果有root密码，则是mysql -u root -p

CREATE DATABASE name; # 新建数据库，name是数据库名字
Query OK, 1 row affected (0.00 sec) #此段为mySQL反馈提示，不需要输入。

CREATE USER user@localhost; # 新建用户，user是用户名字
Query OK, 0 rows affected (0.00 sec) #此段为mySQL反馈提示，不需要输入。

SET PASSWORD FOR user@localhost= PASSWORD("密码"); # 给用户设置一个密码
Query OK, 0 rows affected (0.00 sec) #此段为mySQL反馈提示，不需要输入。

GRANT ALL PRIVILEGES ON name.* TO user@localhost IDENTIFIED BY '密码'; # 把name这
个数据库和user这个用户关联
Query OK, 0 rows affected (0.00 sec) #此段为mySQL反馈提示，不需要输入。

FLUSH PRIVILEGES; # 完成设置
exit # 退出数据库
```

9.2.4 Let's Encrypt，SSL/TLS

http连接，由于不是加密的，所以任何人都可以查看内容，这对于一些金融服务有着巨大的危害，比如使用信用卡在网上购物的时候，账号和密码会被获知。所以网景（Firefox浏览器的前身）提出了SSL（安全套接层/Secure Sockets Layer）这个概念（后来演变升级为TLS，即传输层安全性协议/Transport Layer Security），http变成了https，电脑会内置证书，而网站也会有一个证书，只有两者相互验证成功，才能正常浏览玩网页，并且全程加密（DNS部分并不是加密的，所以有个DoH，dns over https）。

SSL/TLS证书是个垄断行业，电脑内置的证书就那么几家，如果想网站被大多数浏览器/系统接受，那就只能去申请其中某家的证书，这里面层层签发转售，几近无本万利。不过好在还是有很多免费的SSL/TLS证书的，比如Let's Encrypt提供三个月的免费证书，而亚洲诚信通过第三方公司，提供一年免费的证书。这里以Let's Encrypt为例演示，相关链接为 https://certbot.eff.org/instructions：

Let's Encrypt提供的SSL/TLS工具叫做cerbot，可以通过snap或者pip安装。snap是Ubuntu强推的一种软件部署和软件包管理系统，把所有需要的东西都放一起。pip是通过python3的pip安装，pip和snap没有功能上的区别，不想被Ubuntu强推就使用pip。

snap安装cerbot申请SSL/TLS证书

```
apt install snapd
snap install core
snap refresh core
snap install --classic certbot
ln -s /snap/bin/certbot /usr/bin/certbot
certbot --nginx
```

然后按照提示，输入邮箱和同意服务协议，并且在提示域名的时候，注意不要输错。

pip安装cerbot申请SSL/TLS证书

```
apt install python3 python3-venv libaugeas0
python3 -m venv /opt/certbot/
/opt/certbot/bin/pip install --upgrade pip
/opt/certbot/bin/pip install certbot certbot-nginx
ln -s /opt/certbot/bin/certbot /usr/bin/certbot
certbot --nginx
echo "0 0,12 * * * root /opt/certbot/bin/python -c 'import random; import time;
time.sleep(random.random() * 3600)' && certbot renew -q" | sudo tee -a
/etc/crontab > /dev/null
```

相比于snap自动每三个月更新证书，pip需要通过crontab加一个定时任务（上述最后一行），另外，也需要偶尔检查以下certbot有没有更新，即使用此命令`/opt/certbot/bin/pip install --upgrade certbot certbot-nginx`

10 Docker快速入门

Docker的本意是“码头工人”，即搬运别人打包好的集装箱。之所以取这个名字，是因为Docker的功能与此类似：将系统和里面的应用一起打包好，别人“搬走”就能直接使用——可以将Docker粗略的理解成一个包含了系统和应用的虚拟机（严格来说，Docker是使用了沙箱机制的虚拟化容器）。常见的例子就是别人把某一个软件配置好了，用户直接下载下来，简单设置一下就可以使用了，不需要繁杂的配置过程，所以在批量服务中有着广泛的应用场景。Docker分为社区版/CE（Community Edition，免费的）和企业版/EE（Enterprise Edition，收费的），两者功能无本质区别，以下默认使用社区版。

10.1 安装Docker环境

```
apt update # 同步更新源
apt install -y ca-certificates curl gnupg lsb-release # 安装必要依赖软件
```

添加GPG密钥，注意这里和上面一样，必须是root权限（如下命令）。这里简单介绍一下GPG，全称是GnuPG，真·全称是GNU Privacy Guard，一个密码学软件，用来验证通信中的安全性，防止传输过程中被篡改，前身是Pretty Good Privacy/PGP。

```
curl -fsSL https://download.docker.com/linux/debian/gpg | gpg --dearmor -o
/usr/share/keyrings/docker-archive-keyring.gpg
```

选择使用稳定版，如果需要nightly或者test版，可以把下面的stable改成对应的版本。

```
echo "deb [arch=$(dpkg --print-architecture) signed-
by=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/debian $(lsb_release -cs) stable" | tee
/etc/apt/sources.list.d/docker.list > /dev/null
```

更新并安装Docker

```
apt update
apt install -y docker-ce docker-ce-cli containerd.io
```

期间会下载几百兆的文件，网络不好的话，可能会需要一段时间，当完成安装后，使用`docker run hello-world`命令来测试功能是否正常，理论上会输出下图内容：

至此，系统已经安装好了Docker环境，可以自己写一个Docker的应用，或者直接拉取别人写好的为自己所用。

10.2 安装别人打包好的的Docker

先说一下常用的Docker命令，然后以安装Docker版本的Nextcloud为例。Nextcloud是一个开源的网盘系统，类似于私有版本的百度云，可以自己搭建从而确保数据都在自己手上而不会被8秒。

10.2.1 常用Docker命令

`docker ps` 列出所有正在运行的容器，如果需要查看所有的容器（包括未运行或者启动失败的）则使用`docker ps -la`，这点类似于`ls`和`ls -la`的区别。

`docker start/stop/restart CONTAINER ID` 开启/停止/重启特定容器，后面要加上指定的ID，CONTAINER ID见下文。

`docker rm CONTAINER ID` 删除容器，如果是删除镜像，则需要把rm换成rmi

10.2.2 安装Docker版Nextcloud

在 https://hub.docker.com/ 中直接搜索Nextcloud，找到官方版本的镜像，点击进去，在右侧有拉取镜像的命令，直接运行即可。

安装过程中会下载各个组件，等全部显示Pull conplete即表示下载完成，之后会自动校验并提示完成。

使用`docker run -d -p 80:80 nextcloud`运行，此时使用`docker ps`可以查看到具体的详细信息

CONTAINER ID类似于身份证号码；IMAGE是身份证上的姓名；COMMAND是实际运行的程序；CREATED是创建的时间；STATUS是此时的运行状态；PORTS是端口，上述我们把容器的80端口定向到服务器的80，并且默认ipv4和ipv6都可以访问，接受所有IP的访问（0.0.0.0代表接受所有IP）；NAMES是容器的名字，可以理解为外号。

之后就是通过IP或者绑定的域名访问，进行最后的安装。这里就能看出来Docker的优势了：用户无需了解具体操作和搭建步骤，提供者负责维护，这可以极大的简化用户的使用步骤，还可以标准化环境，无论使用Debain还是REHL，镜像/容器都是提供者给定的。

如果不再需要Nextcloud，则首先停止容器，随后再删除：

```
docker stop c30d348f1ef1
docker rm c30d348f1ef1
```

10.3 建自己的Docker

Docker通过Unix socket与它的引擎进行通信，出于安全考虑，一般只有root用户和在docker组的用户才能正常访问Unix socket。所以，如果想建一个完善的Docker应用，那么建议额外新增一个用户，并加入docker用户组。